Offline and Online Power Aware Resource Allocation Algorithms with Migration and Delay Constraints

Khaled Hejja, Xavier Hesselbach

Dept. Network Engineering, Universitat Politecnica de Catalunya, C/ Jordi Girona, 1-3 - Edif.C3 - Campus Nord - 08034 Barcelona - Spain

Abstract

In order to handle advanced mobile broadband services and Internet of Things (IoT), future Internet and 5G networks are expected to leverage the use of network virtualization, be much faster, have greater capacities, provide lower latencies, and significantly be power efficient than current mobile technologies. Therefore, this paper proposes three power aware algorithms for offline, online, and migration applications, solving the resource allocation problem within the frameworks of network function virtualization (NFV) environments in fractions of a second. The proposed algorithms target minimizing the total costs and power consumptions in the physical network through sufficiently allocating the least physical resources to host the demands of the virtual network services, and put into saving mode all other not utilized physical components. Simulations and evaluations of the offline algorithm compared to the state-of-art resulted on lower total costs by 32%. In addition to that, the online algorithm was tested through four different experiments, and the results argued that the overall power consumption of the physical network was highly dependent on the demands' lifetimes, and the strictness of the required end-to-end delay. Regarding migrations during online, the results concluded that the proposed algorithms would be most effective when applied for maintenance and emergency conditions.

Keywords: NFV Resource Allocation; Virtualization; Offline; Online; Migrations; Power Consumption; 5G Delay.

1. Introduction

Future Internet networks and 5G technologies are promising to offer enhanced mobile broadband services, connect massive number of sensors, and natively support mission critical services that require very high reliability and ultra low latency [2]. To realize such networks, multiple design objectives were identified by ITU and 3GPP for 5G including the integration of software defined networking (SDN), networks' virtualization, and network slicing initiatives [3]-[7].

After all, the leading 5G technology that is captivating great deal of attention is network functions virtualization, which separates network functions, such as firewalls, intrusion detectors, and caching, to name a few, from proprietary hardware appliances so they can run in software [7]. However, according to [2] and [6], they stressed on the significance of engineering these NFV based networks to be fully aware about their power efficiency. In practice, this means that NFV infrastructure providers (InP) need to consider the power efficiency of their infrastructure, while virtualizing their physical resources to allocate the demands of the virtual Network Services (NS). In NFV literature, the resource allocation process within NFV architecture is denoted by (RA-NFV) [8][9], in addition to that, virtual Network Services are usually called service function chains (SFC), which are represented as a collection of ordered virtual logical nodes, known as virtual network functions (VNFs) that are connected by a set of virtual logical edges.

On power efficient NFV infrastructure, ETSI in [6] highlighted that the NFV frameworks shall significantly reduce the energy consumption of network infrastructures, so that they support workload consolidations by scaling the traffic loads and concentrate them on a smaller number of servers during off-peak times, and consequently allow to turn-off or put on energy saving mode all other not loaded servers. In accordance with these requirements, several researchers started to apply the consolidation technique to minimize the total power consumption in the physical networks [13]-[15] while allocating the service function chains.

In this paper the authors are extending their previous work from [1] by proposing three new power aware RA-NFV algorithms for offline, online, and migration applications. The main objective of the offline coupled with migration algorithms is to allocate the demands of the Network Services that are known in advance, on the most appropriate physical components that have enough resources, while minimizing the costs of the power consumption in the physical network. The algorithms achieve that through consolidating the network's traffic into least number of active servers as much as possible, then putting other non utilized servers into saving mode. On the other hand, the online with migration algorithms are designed to solve the RA-NFV problem on real time bases and minimize the total power consumptions in the whole physical network. For both scenarios, offline and online, the migration algorithm was kept on alert to be triggered whenever there is a need to transfer some traffic

*Corresponding author

Email address: Khaled.Hejja, Xavier.Hesselbach @ upc.edu (Khaled Hejja, Xavier Hesselbach)

from low utilized resources or during maintenance and emergency conditions.

In general, the algorithms are differentiated from others in literature in the following aspects; first both of them relay on a fast and precise path construction methodology, called segmentation, which was developed by the authors in [1]. It allocates the service function chains, including the demands of their virtual network functions and their edges together and in full coordination on the most fit physical path. Second, up to the time of writing this paper, it is the first time to include migrations and delay in an online power aware algorithm solving the RA-NFV problem. The algorithms solve the RA-NFV problem in fractions of a second, can handle any number of demands, and can be customized to work for any type of physical networks.

Simulations and evaluations for the offline algorithms were conducted against the recent work from [13], focusing on the total power consumption and migration costs of the whole network. Then several experiments were designed to test the online versions developed based on the recent work from [1], covering the overall power consumption performance of the physical network for long periods of time, as well as analyzing the performance of the online algorithm when migrations and end-to-end delay were activated, and analyzing the impacts of SFC's lifetimes on the network.

Main contributions:

1. This paper proposes new offline and online power aware algorithms to solve RA-NFV problem in fractions of a second.
2. Each algorithm is integrated with a migration strategy.
3. To the best of our knowledge, this is the first time to include both migrations and end-to-end delay in an online power aware algorithm to solve the resource allocation problem.
4. All algorithms used a modified version from the path construction methodology by [1] that facilitates allocating precise resources for the virtual network functions and their virtual edges, together and in full coordination on the physical network.

Rest of the paper is organized as follows: Section 2 provides related work, section 3 discusses the offline and online scenarios in high-level, followed by section 4 discussing the modified segmentation methodology used for paths' constructions. Then sections 5 and 6 detail in depth the specifics of the offline and online system models, and the final conclusions are presented in section 7.

2. Related Work

A comprehensive survey of NFV is shown in [8], providing an overview of the key research topics, standardization efforts, early implementation, use cases, and relationship between NFV, SDN and cloud computing. While in [9] the authors presented a classified and comprehensive review for the NFV resource allocation problem, including a classification of the resource allocation problem in NFV, considering the VNFs chain composition, embedding and scheduling, optimization objectives, solution strategies and application domains.

Regarding resource allocation modeling in NFV environments, an early work targeting the complex scheduling problem was presented by [10], in addition to that, the same authors proposed an analytical model for the NFV forwarding graph to optimize the execution time of the Network Services' deployment [11]. An optimal solution for orchestrating the VNFs in small scale networks is presented in [12], who proved the NP-hardness of the NFV orchestration problem, and suggested a heuristic to solve it faster for real-world synthetic topologies and traffic traces.

For the related work covering power efficiency in NFV environment, a recent article by [13] investigated the consolidation, routing and placement problems in NFV architectures based on vertical scaling techniques, and for that the authors proposed a modified algorithm to improve the blocking performance in the scenario in which bandwidth rather than processing capacity is the constrained resource. Most important is that the authors proposed a new consolidation technique taking into account the reconfiguration costs, but with migration strategy. While in [14] the authors formulated and modeled a virtual network function placement problem for power and traffic-aware cost minimization. To solve it, a novel approach that combines the Markov approximation with matching theory was proposed to find an efficient solution for the original problem. Lastly, in [15], the authors proposed a power efficient VNF placement and chaining algorithm that minimizes the power consumption of the servers and switches. They formulated the problem using a Decision Tree model, and solved it using Monte Carlo Tree Search (MCTS) strategy.

Except the work of [13] who included migrations on offline scenarios, all of these recent publications focused on solving the resource allocation problem including power optimization, but without migrations. In addition to that, and up to the time of publication of this paper, no literature could be found solving the resource allocation problem while considering power minimization in online with migration and delay.

Regarding related work about delay and NFV, the authors in [16] proposed a VNF placement algorithm that allows users to meet their service latency requirements while minimizing the power consumption at the same time. In the proposed algorithm, a shortest path between a source and a destination is first computed, using Dijkstra algorithm based on the service latency requirements from the users. Then, the VNFs are allocated to the appropriate virtual machines (VMs) representing the physical infrastructure component along the path that can minimize the power consumption.

In [17], the authors tried to find the most suitable node for the placement of a VNF from the service chain in order to minimize the total power consumption cost with certain constraints. They designed a power saving model using queuing network for the placement of multiple service chains on the network, and used delay as a constraint. Moreover, [18] developed an optimization model to solve the chaining problem including end-to-end delays, while [19] generalized an optimization model to cope

with the arbitrary VNF chains and traffic demand uncertainties, fulfilling individual average round trip delay constraint for each VNF chain.

Moreover, [20] formulated the delay-aware virtual network function scheduling problem, considering both VNFs' processing delays at VMs, and service chain transmission delays at virtual links, and to solve the problem, the authors developed a heuristic using genetic algorithm to overcome the complexity of the original problem. Finally, in [21], the authors presented an analytical model for the placement of service function chains in multi cloud environments, and tried to optimize the end-to-end delays to end users with optimal placements of the service function chains in a multi cloud scenario.

3. Offline and Online Scenarios

The algorithms proposed by this paper were designed to work for offline and online scenarios. In offline, all SFCs are assumed known in advance, while in the online scenario they are assumed arriving on real time, and each has arrival and lifetime. The following paragraphs introduce the overall idea of each scenario as follows:

3.1. Offline Scenario

The offline scenario is designed to evaluate the overall performance of the proposed algorithms under fully controlled conditions when allocating the service function chains. In this paper, the offline algorithm from [1] is modified to work on NFV framework, and accordingly, two algorithms were proposed for the offline scenario, the first one is an offline power aware coordinated NFV algorithm, PaNFV, coordinating the allocations of the physical servers and edges' resources as demanded by the virtual nodes and edges of the service function chains, and the second algorithm is the migration extension of PaNFV, denoted by mPaNFV, which is triggered by the offline algorithm, PaNFV, to check if migrating some or all of the already allocated SFCs would result on minimizing the total costs of the physical infrastructure network.

3.2. Online Scenario

It represents real life conditions where the SFCs' arrival times and lifetimes are random and not known in advance, which means that the online algorithm has to deal with each SFC instantly and one-by-one. In the following paragraphs, a summery for some of the related work that focused on online resource allocation and migration strategies will be introduced.

For example, the recent work by [25] proposed a user distribution aware virtual machine redeployment mechanism and proposed a traffic redirection virtual machine migration scheme to keep the active flows from being interrupted. However, the authors did not evaluate the impact of their work on the power consumption of the network, and they did not include end-to-end delay as a main constraint.

Moreover, the authors in [26] considered rack, cooling structure and network topology when consolidating virtual machines inside a datacenter. They provided a dynamic placement method considering the utilization of racks, in addition to utilization of individual servers to improve the energy consumption in the cooling system and the datacenter network. They considered migrating the virtual machines of underutilized racks onto other utilized racks and turned off the idle racks in order to save more power. However, the authors did not consider resource allocations for distributed datacenters, and did not consider the impact of delay in their work. Another study was conducted by [27] which focused on improving the virtual machines' placement problem inside a datacenter, by combining virtual machines for placement based on predicting the required processing power and bandwidth resource demands in order to reduce the amount of resources, and to improve both energy efficiency and performance. However, their work also did not include distributed datacenters, it treated migrations of individual virtual machines and not the network service as whole, and they did not consider end-to-end delay as a constraint in their work.

In [28] the authors proposed a reinforcement learning algorithm that can determine the optimal action from a set of actions. The proposed algorithm considers changes in the demanded resources to predict the physical machine which may suffer from overload. To improve the utilization and to reduce the energy consumption in the datacenter, the proposed algorithm will turn off the idle physical machines after migrating their traffic to other machines. However, the proposed algorithm by the authors was designed for large datacenter, and did not include distributed networks of datacenters as well as not including end-to-end delay. Finally, the authors in [29] proposed an energy aware dynamic optimization approach using artificial bee colony, to find the mapping relation between physical machines and virtual machines inside a datacenter. The algorithm constrains the consolidations in a manner that limits the number of virtual machines' migrations, to minimize the negative impacts on quality of service. In addition to that, the algorithm attempts to minimize overloading the physical machines, and turns off underloaded physical machines to reduce the energy consumption of the datacenter. Also in this work, the authors did not consider distributed datacenters, and end-to-end delay as a resource allocation constraint too.

Accordingly, for the online scenario this paper proposes a modified version of the online algorithm from [1], which included online resource allocations, and considers end-to-end delay as a main allocation constraint. The proposed new algorithm is called, the online power aware coordinated NFV algorithm, oPaNFV, which ensures performing the allocation process based on minimizing the total power consumption in a distributed network of datacenters. It allocates the arriving SFCs once they arrive the physical network on real time, and terminates them from the physical network once their lifetimes get expired. oPaNFV treats the SFCs as first-arrived-first-served without any additional filtering, applies virtual machines consolidations to minimize total power consumption in the datacenters' network, and ensures selecting the physical path with the least end-to-end delay to meet the requirements of the SFCs.

Moreover, this paper integrates the migration algorithm mPaNFV with oPaNFV to perform the migrations of the SFCs within a network of distributed datacenters for the online sce-

nario, and it also considers the end-to-end delay while migrations to meet future 5G core networks.

4. Understanding Segmentation Technique

In this paper the segmentation technique developed by [1] is modified to work for an NFV based physical networks when constructing the candidate paths for allocating the service function chains. Therefore, the following paragraphs discuss in depth the segmentation technique for datacenters using NFV architecture, in addition to explaining how to construct the segments of a service function chain and a physical path.

4.1. Overview

Assume that the physical network is composed of datacenters having a collection of access, router, switch, and server nodes, connected by physical edges as the example shown in Fig.1b. Also assume that the edges connecting any pair of nodes in the physical network are fixed and can be identified in advance including their parameters. Moreover, assume that the Network Service (i.e SFC) is represented as a graph composed of source and termination access nodes, and orderly chained virtual network functions as shown in the example in Fig.1a. To solve the resource allocation problem, the path construction methodology from [1] is applied to guarantee the coordinated allocations of the virtual functions and their connecting virtual edges, together, on a physical path between the datacenters that can fulfill the demands of the SFC. Each path could be constructed of one or multiple segments, giving that each segment represents a pair of nodes and their edge.

4.2. Definitions

4.2.1. Segmentation concept and notations

Segmentation concept is a representation of the information in any path. Assume that the set R of SFCs are required to be allocated on the physical network of datacenters, the proposed segmentation methodology will start converting the structure of the SFC number $r \in R$, listing all information about its nodes and edges together in a set format denoted by Seg^r, which represents the demands of SFC^r. Afterwards, to allocate this SFC^r, the information about the resources of a specific physical network path, including its nodes and edges, will also be converted into a physical segment format as well, and will be denoted as Seg^P.

4.2.2. General definition of the basic segment

For any pair of nodes, a and b, and their edge (a, b), a basic segment lists the nodes' parameters such as their locations, a_{loc} and b_{loc}, processing powers of the nodes, cpu_a and cpu_b, and the edge's parameters such as, bandwidth bw_{ab}, and delay d_{ab}, all to be grouped in one set format denoted by Seg_{ab} as shown in Eq.(1).

$$Seg_{ab} = \{a_{loc}, b_{loc}, cpu_a, cpu_b, bw_{ab}, d_{ab}\} \tag{1}$$

4.2.3. Segmentation for a path of a single basic segment

Assume a path P_{ij} composed of two access nodes, source i and destination j, reaching node s, which could be a server if the path is inside a datacenter, or s could be a VNF if the path represents an SFC. Consequently, path P_{ij} segment denoted by Seg^P_{ij} is defined as a list of the basic segments $Seg_{i,s}$ and $Seg_{s,j}$, including all parameters of their nodes and edges grouped in a set format as shown in Eq.(2). It includes locations of the source and destination nodes, denoted by i_{loc} and j_{loc}, processing power capacities of node s in that path, cpu_p, and the parameters of the connecting edges, including bandwidth capacities, bw_{is}, bw_{sj}, and delay d_{ij}. Important hint, in this paper, it is assumed that only the server nodes have computing resources, but access, router, and switch nodes do not.

The segment defining path P_{ij} can be written in set format as follows:

$$Seg^P_{ij} = \{i_{loc}, j_{loc}, cpu_p, bw_{is}, bw_{sj}, d_{ij}\} \tag{2}$$

4.3. Application of Segmentation on NFV framework

Let the network shown in Fig.1b be used to generalize the concept of a single segment path shown in Eq.(2) on paths composed of multiple basic segments. Moreover, assume the network is using NFV framework, separating control plan from data plan, and the physical network includes datacenters (Datacenter-1 and Datacenter-2) of fat-tree topology, composed of four access nodes, a, b, c, d, two routers assigned as nodes 1 and 2, two switch nodes 3 and 4, and five server nodes 5, 6, 7, 8, and 9. The physical server nodes are identified by their processing cores, cpu_p, and the physical edges by their bandwidth capacities, bw_{ij}, and the delay in each of them, d_{ij}. Finlay, assume that the VNFs constructing any SFC are of different types and the servers can treat anyone of them without any restrictions.

To construct the segments, following paragraph will introduce how to build the service function chain segment, Seg^r, and based on that, the next paragraph will explain the segment formulation for a physical path, Seg^P_{ac}, matching the locations of the access nodes in Seg^r.

4.3.1. Formulating service function chain segment (Seg^r)

Assume SFC$^r_{ac}$ demanding allocating two different types of VNFs, VNF^1 of core processing power cpu^r_1, and VNF^2 of cpu^r_2. Moreover, assume that SFC$^r_{ac}$ demands to forward throughput capacity of bw^r at delay rates not exceeding d^r between access nodes a (at Datacenter-1) and c (at Datacenter-2). To construct SFC$^r_{ac}$ segment, the concept of path segment in Eq.(2) can be generalized on the graph of SFC$^r_{ac}$ shown in Fig.1a, and the resulting segment is formulated as in Eq.(3) bellow:

$$Seg^r_{ac} = \{a^r_{loc}, c^r_{loc}, cpu^r_1, cpu^r_2, bw^r, d^r\} \tag{3}$$

In this way, the demands of SFC$^r_{ac}$ are translated into a segment representing the demands of SFC$^r_{ac}$ as illustrated in Fig.1b.

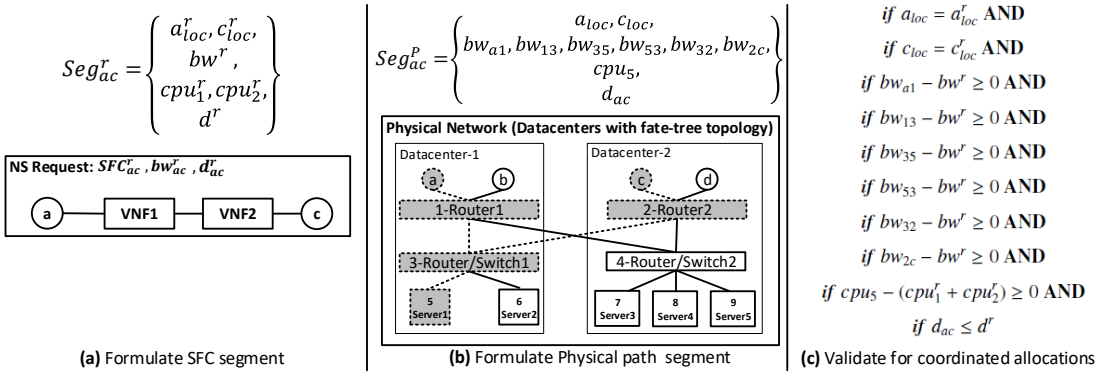

Figure 1: Segments' formulation demonstration: (a) shows the service function chain SFC^r_{ac} and at the top its segment, (b) shows Physical network and the segment of a candidate hosting path P_{ac}, and (c) shows the (if-AND) statements to validate the fully coordinated allocation of SFC^r_{ac} nodes and edges on P_{ac}.

4.3.2. Formulating physical path segment (Seg^P_{ac})

Assume that the NFV orchestrator has already the list of all pairs in the physical network shown in Fig.1b, and it can instruct the hypervisor layer (using the algorithms proposed by this paper) to list all pairs that can be used to construct each physical path in the network in advance. Hence, the segmentation section in the algorithm will select a path fulfilling the locations of the demanded access nodes and all the pairs in between. To construct the path, it will connect the last node in the first pair with the first node in the next pair, if there is an edge between them, otherwise, it will go to the next pair, and do the same to the remaining pairs in the list of that path, until all pairs are connected properly formulating the path and its segment. Ultimately, the proposed algorithms residing at the hypervisor layer will follow this same methodology to construct and list all paths in the physical network.

To clarify that, refer to Fig.1b. Given that all physical paths and the list of their pairs are known in advance, and since SFC^r_{ac} demanding access between nodes a and c, the proposed allocation algorithm will select the physical paths that match the demanded source and destination access nodes, then it will rank them according to the consumed cores of the servers in these paths. To test the possibilities of allocating SFC^r_{ac}, it will select the top ranked physical path, and formulate its segment.

For example, assume the physical path $P_{ac} = \{a, 1, 3, 5, 3, 2, c\}$ was the top ranked path, starting from the demanded source node a passing through router-1 (defined as node 1), then switch-1 (as node 3), reaching server-1 (as node 5). This is the forward path from source access node a to server node 5 given as $\{a, 1, 3, 5\}$. The other forward path $\{5, 3, 2, c\}$ from server 5 to destination access node c, starts from server node 5, passing through switch-1 (node 3), then router-2 (as node 2), and finally reaches the demanded destination access node c. Accordingly, Seg^P_{ac} representing path P_{ac} can be constructed as shown in Eq.(4):

$$Seg^P_{ac} = \{a_{loc}, c_{loc}, bw_{a1}, bw_{13}, bw_{35}, bw_{53}, bw_{32}, bw_{2c}, cpu_5, d_{ac}\} \quad (4)$$

Given that:

$$d_{ac} = d_{a1} + d_{13} + d_{35} + d_{53} + d_{32} + d_{2c}$$

4.4. Segmentation technique for coordinated allocation

To allocate all demands of SFC^r_{ac} on the physical path P_{ac} in full coordination, allocating the virtual nodes together in the same step with the allocation of the virtual edges, each element in the two formulated set of segments Seg^r_{ac} and Seg^P_{ac} will be directly compared one-to-one, which means for each parameter value in Seg^r_{ac} representing a virtual function in SFC^r_{ac}, check if its counterpart in Seg^P_{ac} can fulfill its demands, and for the demanded bandwidth capacity and end-to-end delay by Seg^r_{ac}, check if each edge in Seg^P_{ac} has at least enough residual capacity to host that bandwidth without exceeding the demanded delay threshold.

This process can be achieved ***if and only if*** each comparison statement in the (**if-AND**) conditions shown in Fig.1c is true, which guarantees that all the demands of SFC^r_{ac}, including access nodes locations, processing powers of its virtual functions, bandwidth, and end-to-end delay, will be hosted together as one set by their counterparts in the physical path P_{ac}.

Therefore, segments formulation based on Eq.(2), and the check-up comparisons, made the allocations process of the SFCs' virtual functions, fully coordinated with the allocations of their virtual edges, together, on the same physical path.

5. Offline scenario, modeling, algorithm, and simulation

The following subsections will introduce and explain the overall design model for the RA-NFV system in offline scenario, starting by defining the physical network model and the service function chain model, then introduce the offline problem formulation including the objective function and its constraints. Moreover, the offline resource allocation and migration algorithms will be explained and discussed with a simple example, and the final paragraph in this section will introduce the offline simulation settings and discuss the final results in more details. Notations and their description are summarized in Table 1.

5.1. Physical Network Model

The physical network is assumed following the network function virtualization architecture as proposed by [7]. It is modeled as a weighted directed graph $G^P = (N^P, E^P)$, where N^P and E^P are the sets of physical nodes and edges respectively, given that $i \in N^P$ represents a physical node from the set N^P, and $(i, j) \in E^P$ represents a physical edge from the set E^P. The set of physical nodes N^P is composed of two sets, $i^a \in N^P_A$ referring to the physical access nodes, which has no computing resources, but only serves as an originating or terminating interface points for the service function chains SFCs, and $i^p \in N^P$ referring to the set of physical server nodes, which will host the virtual functions of the SFCs. Each $i^p \in N^P$ is characterized by the following parameters: maximum processing power capacity CPU^p_i in terms of the number of cores in i^p, cpu_{i^p} representing the current available cores, cpu^{min} the minimum consumed cores to trigger traffic migration, and P^{Busy} and P^{idle} are the maximum and idle power consumptions of the physical server. Each physical edge $(i, j) \in E^P$ transferring the traffic between a pair of physical nodes i and j is associated with maximum bandwidth capacity given by, BW_{ij} in bps, bw_{ij} current available bandwidth, and current propagation delay given by d_{ij} in milliseconds. In addition to that, the network is also characterized by the set of all directed paths given by $P^P = \{P_{sd}\}$, where $P_{sd} = \{(i, j)\}$ represents a directed path connecting any physical source node s to destination node d and is constructed of more than one physical edge (i, j).

5.2. Service Function Chain Request Model

In this paper, the Authors formulated the SFCs segments in a very generic structure, so they can handle any type of VNFs. Accordingly, the ILP model does not include any constraints about the type of VNFs. Therefore, the SFCs model considers the set F, representing all types of virtual network functions that can be used to compose logical graphs representing each SFC, where $F = f_1, f_2, ..., f_F$ and f_u being the u^{th} VNF type (typically a VNF could be: Deep Packet Inspector (DPI), Policy Control and Charging Rules Function (PCRF), Load Balancer (LB), Firewall (FW),...), and each has a specific packet processing time given by t^{proc}_u. Assume it is required to allocate a total of R service function chain requests, where SFC^r is composed of one or more f_u nodes and can be modeled as a weighted and directed graph $G^r = (N^r, E^r)$, where N^r and E^r are the sets of the logical f_u nodes and their connecting logical edges respectively. $u \in N^r$ is the VNF node u in the set of VNFs N^r, and $(u, v) \in E^r$ is a virtual logical edge belonging to the set of edges E^r connecting VNFs u and v. The set of VNFs N^r are composed of two sets, N^r_A representing the set of fictitious logical access nodes used for originating and terminating SFC^r, and N^r representing the set of logical VNF server nodes belonging to SFC^r, and $u \in N^r$ is a logical virtual server node associated with cpu_u representing the demanded cores for consumption by VNF node u, and T^u is the sum of throughputs incoming to VNF node u. Each SFC^r is associated with L^r representing the traffic flows' packet length of SFC^r, demanded bandwidth capacity denoted by bw^r, and d^r is the maximum allowed end-to-end delay by SFC^r.

5.3. Offline problem Formulation

In this paper the RA-NFV problem is modeled as an integer linear programming (ILP) problem of optimization objective function, having positive integer and linear variables, and solved based on satisfying certain number of constraints. The following paragraphs introduce and formulate the RA-NFV ILP objective function and its constraints as follows:

5.3.1. Offline RA-NFV objective function

For the offline scenario, the objective function aims to minimize the total cost C^{tot}, which sums the total power consumption costs, C^{pc}, and the total migration costs, C^{mig}, in the whole physical network when S_R successful SFCs' allocations were performed [13]. The power consumption costs, C^{pc}, sums up the cost of the consumed power by the servers when they were idle, and the cost of the traffic handled by the servers due to the allocated SFCs. The migration costs, C^{mig}, includes the impacts due to migrating all or some of the S_R successfully allocated SFCs. It is important to notice that the migration costs reflect the allocations' efficiency, giving that the lower the migration costs the more optimal and efficient were the allocations in the first place.

To make sure that a specific server node is hosting at least one VNF, variable $x^u_{i^p}$ is used in the ILP objective function formulation, which takes a binary value of 1 if server node i^p is assigned to host VNF u from SFC^r. Variable λ_{i^p} is 1 if physical node i^p is active, or 0 if it is turned-off, variable $mig^u_{i^p}$ is 1 if VNF u hosted by i^p is migrating, or 0 if not. Variable $mig^{uv^r}_{ij}$ is set to 1 if virtual edge (u, v) hosted by the physical edge (i, j) is migrating, variable mig^r is set to 1 if SFC^r is migrating, and variable $x^{uv^r}_{ij}$ is set to 1 if physical edge (i, j) is hosting virtual edge (u, v).

Mathematical formulation of the objective function C^{tot}, C^{pc}, and C^{mig} were based on [13] and are presented in Eq.(5), Eq.(6), and Eq.(7) as follows:

$\forall i^p \in N^P,\ \forall r \in R$

$$min\ C^{tot} = C^{pc} + C^{mig} \tag{5}$$

$$C^{pc} = \beta_e \Delta t (P^{idle} \sum_{i^p \in N^P} \lambda_{i^p} + \sum_{i^p \in N^P} \frac{\sum_{u \in N^r} x^u_{i^p} t^{proc}_u T^u / L^u}{CPU_{i^p}}) \tag{6}$$

$$C^{mig} = \beta_d \sum_{r \in S_R, u \in N^r} T^r_{down} T^u \mu^u_{i^p} \tag{7}$$

β_e is the unit cost of a consumed 1 Watt of power, β_d is the revenue loss due to migrating 1 Gbit of traffic, Δt is the duration of a cyclic stationary interval, T^r_{down} is the downtime of all VNFs in SFC^r during the migration process, and T^u is set equal to the demanded bandwidth by the SFC, since each VNF has one incoming edge only [13].

Accordingly, to minimize the overall costs in the physical network, the objective function works in two directions: first it attempts to minimize the total power consumption costs in the whole physical network due to all allocated SFC^r. Second, to further minimize the costs of the physical network, the objective function considers the migration costs coming from migrating

Table 1: Notation

Notation	Description.	Notation	Description.
G^P	Physical network directed graph.	N^P	Set of all physical nodes.
N_A^P	Set of all physical access nodes.	E^P	Physical network edges.
CPU_{i^p}	Maximum CPU capacity at physical server i.	cpu_{i^p}	Current available CPU at physical server i.
cpu^{min}	Minimum CPU to trigger migration.	P^{idle}	Idle power consumption of the server node.
P^{Busy}	Maximum power consumption of the server node.	BW_{ij}	Maximum bandwidth of edge (i, j).
bw_{ij}	Current available bandwidth in edge (i, j).	d_{ij}	Current delay in physical edge (i, j).
P^P	All paths in the physical network.	P_{sd}	Physical path between s and d.
F	All virtual network functions' types.	SFC^r	Service function chain number r.
G^r	SFC^r weighted directed graph.	N^r	All VNFs in SFC^r.
N_A^r	Set of all logical access nodes.	E^r	All virtual edges in SFC^r.
T_{down}^r	Average down time of migrating SFC^r.	R	Set of SFCs to be allocated.
Δt	Cyclic stationary intervals' duration.	T^u	Sum of throughputs incoming to VNF u.
L^u	Packet length of SFC's traffic flows in Bytes.	cpu_u	Demanded CPU capacity by VNF u.
t_u^{proc}	Packet processing time of u.	bw^r	Demanded bandwidth by SFC^r.
d^r	Demanded maximum end-to-end delay by SFC^r.	S_R	Set of successfully allocated SFCs.
Seg^P	P_{sd} segment listing all its parameters.	Seg^r	SFC^r segment listing all its parameters.
C^{tot}	Total cost of the physical network.	C^{pc}	Cost of power consumption.
C^{mig}	Cost of migration.	$x_{i^p}^u$	1 if i^p is hosting VNF u.
λ_{i^p}	1 if i^p is active.	$mig_{i^p}^u$	1 if VNF u hosted by i^p is migrating.
$mig_{ij}^{uv^r}$	1 if (u, v) hosted by (i, j) is migrating.	mig^r	1 if SFC^r is migrating.
$x_{ij}^{uv^r}$	1 if (i, j) is hosting (u, v).		

the hosted traffic by the under utilized physical servers to other more utilized physical servers. This in turn should lead to freeing and turning off the old servers if any.

5.3.2. Constraints' definition and formulation

The solution of the objective function Eq.(5) will be controlled by capacity and domain constraints as shown bellow.

1. *Constraints on the physical server nodes*

To ensure that the maximum CPU capacity in physical server node i^p is greater than or equal, to the demanded capacities by all allocated VNFs on this server node, Eq.(8) is formulated as follows:

$$\forall\ i^p \in N^P \sum_{r\in R, u\in N^r} cpu_u x_{i^p}^u \leq CPU_{i^p} \tag{8}$$

To ensure that a server is switched-on when it hosts at least one virtual function node Eq.(9) is formulated as follows:

$$\forall\ i^p \in N^P, \sum_{u\in N^r} x_{i^p}^u \geq \lambda_{i^p} \tag{9}$$

2. *Constraints on the physical edges*

To ensure that the total consumed bandwidth on physical edge (i, j) is less than or equal to the maximum bandwidth capacity at that edge, Eq.(10) is formulated as follows:

$$\forall\ ij \in E^P\ \forall\ uv \in E^r \sum_{r\in R} bw^r x_{ij}^{uv^r} \leq BW_{ij} \tag{10}$$

To ensure that the current end-to-end delay in the physical path P_{sd} is less than or equal to the maximum allowed delay d^r by SFC^r, Eq.(11) is formulated as follows:

$$r \in R \sum_{\forall\ (i,j)\in P_{sd}} d_{ij} \leq d^r \tag{11}$$

3. *Migration constraints*

To ensure that a virtual function node u is migrating, if the processing capacity of the physical server node i^p hosting it is below the threshold value cpu^{min}, Eq.(12) is formulated as follows:

$$\forall\ i^p \in N^P \sum_{r\in R, u\in N^r} cpu_u x_{i^p}^u \leq cpu^{min} \tag{12}$$

To ensure that the maximum CPU capacity in physical server node i^p is greater than or equal to the demanded capacities by the already allocated VNFs v plus the migrating VNFs u on this server node, Eq.(13) is formulated as follows:

$$\forall\ i^p \in N^P \sum_{r\in R, v\in N^r} cpu_v x_{i^p}^v + \sum_{r\in R, u\in N^r} cpu_u mig_{i^p}^u \leq CPU_{i^p} \tag{13}$$

To ensure that the maximum BW capacity in physical edge (i, j) is greater than or equal to the demanded BW capacities by the already allocated edges (w, z) plus the migrating edges (u, v), Eq.(14) is formulated as follows:

$$\forall\ ij \in E^P \sum_{r\in R, \forall\ wz\in E^r} bw^r x_{ij}^{wz^r} + \sum_{r\in R, \forall\ uv\in E^r} bw^r mig_{ij}^{uv^r} \leq BW_{ij} \tag{14}$$

To ensure that the current end-to-end delay in the physical path P_{sd} is less than or equal to the maximum

allowed delay d^r by the migrating SFC^r, Eq.(15) is formulated as follows:

$$r \in R \sum_{\forall\ (i,j) \in P_{sd}} d_{ij} \leq d^r mig^r \tag{15}$$

4. *Domain constraints*

To solve the problem as an ILP, Eq.(16)-Eq.(19) are defined as follows:

$$\forall\ i^p \in N^P\ x^u_{i^p} \in \{0,1\} \tag{16}$$

$$\forall\ i^p \in N^P\ \mu^u_{i^p} \in \{0,1\} \tag{17}$$

$$\forall\ i^p \in N^P\ \lambda_{i^p} \in \{0,1\} \tag{18}$$

$$\forall\ ij \in E^P\ \forall\ uv \in E^r\ x^{uv^r}_{ij} \in \{0,1\} \tag{19}$$

To ensure that all virtual functions $u \in N^r$ of a single SFC^r are mapped only to one physical server node i^p, Eq.(20) is defined as follows:

$$\forall u \in N^r \sum_{\forall i^p \in N^P} x^u_{i^p} = 1, \tag{20}$$

5.4. PaNFV to solve RA-NFV in full coordination

Optimal solution to solve the RA-NFV objective function in Eq.(5) under the constraints in Eq.(8)-Eq.(20) implies allocating the virtual functions on the physical server nodes that are capable of meeting the demanded computing resources. Theoretically, this is used to be performed through introducing binary constraints to allocate all VNFs belonging to SFC^r on one physical server node, which is similar to the multi-dimensional Bin Packing problem [13]. Moreover, the optimal solution for Eq.(5) implies also connecting one edge only for each server node, and this is usually treated as a commodity between pairs of nodes, which is similar to finding an optimal flow for the commodity in any network model, and that was proved to be an NP-hard problem and not solvable in polynomial times [24].

Consequently, the majority of RA-NFV approaches followed heuristic or meta-heuristic algorithms to solve the optimization problem in a reasonable polynomial time [8][9]. Therefore, this paper proposes a new RA-NFV power aware algorithm, PaNFV, to solve the offline objective function by fully coordinating allocating the VNF nodes and edges together, while minimizing the power consumption in the network as much as possible. PaNFV applies the segmentation approach for constructing the best path, which will be used to allocate the virtual functions and edges as will be explained in the following paragraphs.

5.5. PaNFV code explained:

Pseudo-code for PaNFV heuristic is shown in Algorithm 1, which is explained as follows:

5.5.1. Initialization

The PaNFV is designed to work on NFV environments, giving that the physical network is constructed of an interconnected datacenters using access, router, and switch nodes, and each datacenter hosts a collection of physical servers in the form of fat-tree topology. The access, router, and switch nodes do not have any processing cores, only the servers will be characterized by specific number of cores, and each edge connecting these nodes has specific bandwidth and delay values. The algorithm starts listing all physical paths connecting source-to-server nodes, as well as the paths from server-to-destination nodes. Its assumed that the physical network is fixed, therefore, the main elements formulating any path, such as number and connectivity of the nodes and edges are also fixed and do not change, but only their capacities varies due to the consumption.

Algorithm 1, PaNFV Pseudo-Code

1. Input: G^P and G^r.
2. **For** each SFC^r formulate SFC^r parameters into segment Seg^r generalizing Eq.(2).
3. **For** the set of all saved physical paths P^P **Do**
   - **List** all physical paths matching the sources and destination access nodes as requested by SFC^r.
   - **Rank** them in descending order based on the consumed cores of the server nodes in these paths.
4. **For** the top ranked path P_{sd}, formulate its segment Seg^P according to Eq.(2).
5. **Compare** each element in Seg^r against its counterpart in Seg^P
   - **Check** for CPU, BW and Delay constraints according to Eq.(21).
6. **If** satisfied, allocate SFC^r on P_{sd}.
   - **For** P_{sd} server node and edges, update CPU and BW resources.
   - **Else** go to next ranked physical path, step-3.
7. **For** all idle server nodes, turn them off to save power.
8. **For** all active server nodes **Do**
   - **If** Traffic at low profile, or consumed cores in the server are less than or equal to the cpu^{min} **Do**
     - **Call the migration code mPaNFV**
   - **Else** Continue
9. Calculate the concerned metrics.
10. **If** SFC^r list is not empty **- Go** to next SFC^r step-2.
11. **End**

The initialization step is performed in advance and ahead of handling any SFC, accordingly, the PaNFV algorithm will always have a full list of all the paths in the network, as well as a detailed information about the nodes and edges of these paths, such as, number of nodes and edges in the path, types of the nodes (access, router, switch, or server), consumed capacities from the resources of server nodes or edges, as well as end-to-end delay per each edge in the path.

5.5.2. Segmentation and ranking

In this step, PaNFV algorithm formulates SFC^r segment Seg^r following the format of Eq.(3), then to formulate the physical

segment Seg^P of a candidate physical path P_{sd} it recalls all physical paths that start and terminate with the access nodes that were requested by the SFC^r. Next, it ranks these paths according to the utilization of their servers, which is a fraction between (0-1) calculating the ratio of consumed cpu_{i^p} to the maximum CPU_{i^p}. Accordingly, PaNFV selects the top ranked physical path P_{sd} and formulates its segment following the format of Eq.(4), and handover Seg^r and Seg^P to the allocation stage.

5.5.3. Allocation decision

Similar to paths ranking, PaNFV can rank the SFCs based on their demanded processing powers, throughput, delay, and also can order them according to their revenue generation potentials. However, to speed up the allocation process, this paper processes the SFCs based on first listed first allocated. Consequently, the offline algorithm, PaNFV, lists all SFCs without any ordering, and then it handles each SFC starting from the top of the list, and proceeds to allocate it. Once an SFC was successfully allocated, it directly moves to the next SFC in the list of SFCs to be allocated.

Consequently, to guarantee full coordinated allocations for all VNFs and virtual edges in SFC^r, PaNFV compares each element in the physical segment Seg^P to its counterpart in Seg^r, one-to-one at the same time and in one comparison step as shown in the comparison inequalities given by Eq.(21).

Allocating virtual network functions. Generally, PaNFV can allocate the VNFs of any SFC on one server, or distribute them among multiple servers. However, since the main objective of PaNFV is to minimize the total costs, allocating all VNFs of an SFC on only one single server, will minimize the total costs as low as possible, by utilizing all active servers to their full capacities before activating new ones. This will allow the PaNFV algorithm to use the least physical resources, and efficiently utilize the network resources to their full capacities. Moreover, allocating all VNFs of any SFC on one server will help in speeding up the migration processes, minimize impacts of delays, and increase service quality.

Accordingly, when allocating the VNFs of any SFC on a single server, PaNFV can work in two modes:

1. The original basic mode called (consolidated VNFs) that allocates all demanded cores on the same server, one after another. This is used in case the VNFs were not ordered in the SFC, and for strict quality of service conditions. The expense of using this mode is going to be consuming more resources from the servers.
2. The flexible mode called (non-consolidated VNFs), which allocates cores on the physical server based on the total demanded cores by the largest VNF in the SFC. This is used in case the VNFs are strictly ordered, and to utilize the servers to their maximum capacities.

Accordingly, if consolidated mode is allowed, cpu^u in Eq.(21) will be the sum of the demanded cores by all VNFs in the SFC, or if non-consolidated mode is allowed cpu^u will be represented by the demanded cores of the largest VNF in the SFC.

Allocating virtual edges. PaNFV checks if the available bandwidth in each edge of the physical segment Seg^P is at least equal to the demanded bandwidth element in Seg^r. bw^r in Eq.(21) is the bandwidth of the minimum edge in the SFC, and for delay, PaNFV sums the current delays in all edges of P_{sd} and compares that to the demanded d^r in Seg^r.

Notice the use of $'$**AND**$'$ in Eq.(21) to guarantee the full coordination, which is the tool that PaNFV uses to ensure that all allocations will be fulfilled ***if and only if*** every element in Seg^P satisfies the needs of its counterpart in Seg^r. Therefore, if the inequalities in Eq.(21) are '**ALL**' true, this means that the demands of SFC^r virtual function nodes and edges can be accommodated by the physical path represented by Seg^P. Consequently, a successful allocation is performed for both, VNFs and virtual edges of SFC^r together on the path of the physical segment Seg^P.

Decision matrix for the allocation process is given as follows:

$$\begin{gathered} \forall\ i^p \in P_{sd}\ and\ \forall u \in N^r\ \textbf{\textit{if}}\ cpu_{i^p} - cpu_u \geq 0\ \textbf{AND} \\ \forall\ ij \in P_{sd}\ and\ \forall r \in R\ \textbf{\textit{if}}\ bw_{ij} - bw^r \geq 0\ \textbf{AND} \\ \textbf{\textit{if}} \sum_{ij \in P_{sd}} d_{ij} \leq d^r \end{gathered} \tag{21}$$

5.5.4. Updating

Once a successful allocation occurs, the algorithm updates all changed resources and moves to the next SFC^r. However, in case that the candidate physical segment does not have enough resources to accommodate the demands of SFC^r, the algorithm jumps to the next ranked physical path and follows on from step-3. This process keeps on going until no more SFC^r to be handled.

5.6. mPaNFV code explained

The PaNFV triggers the migration phase and calls mPaNFV at two conditions, 1) if the traffic level in the network is very low (i.e, off peak-times), or 2) at any time it detects the utilization of any active server node is below or equal to the migration threshold defined by parameter cpu^{min}. The migration's pseudo-code is shown in Algorithm 2, where mPaNFV lists all allocated SFC^{r}s in a list of candidates for migration and at the same time it lists their hosting physical paths, then it sorts and ranks all these physical paths according to the utilization of their server nodes.

Next, it performs three (if) conditional checks, starting by the top ranked physical path and the first SFC^r in the candidates for migration list, if (first condition) the top ranked physical path is currently hosting this candidate SFC^r no migration occurs, and it jumps to the next SFC^r in the list. Otherwise, mPaNFV goes to check if (second condition) the server in the top ranked physical path is different from the server currently hosting SFC^r, if not true it jumps to the next ranked path, otherwise it checks (third condition) if the utilization of the server in the top ranked physical path is higher than the utilization of the current server hosting SFC^r.

If all checks were true, mPaNFV compares the segment of the selected top ranked physical path Seg^P to the segment of

the candidate migrating SFC^r Seg^r, if the new physical path has enough residual resources to host the migrating SFC^r, mPaNFV performs the allocation, updates the CPU and BW of the old and new physical paths, then it turns off the old server node to minimize the power consumption. However, if the new physical path does not have enough resources to host the migrating SFC^r, mPaNFV jumps to the next ranked physical path. On the other hand, if there was no physical path to migrate the SFC^r to it, the mPaNFV jumps to the next SFC^r in the list for migration. Once there are no more SFC^r to migrate, mPaNFV returns again to the allocation algorithm PaNFV to continue allocating other SFC^r.

Algorithm 2, mPaNFV Pseudo-Code

1. **Input:**
   - List all allocated SFC^r.
   - List all physical paths that are currently hosting SFC^{r}s.
2. **For** each allocated SFC^r, reformulate its Seg^r Eq.(2).
3. **- List** all physical paths already in use, matching the source and destination access nodes as requested by SFC^r.
   - Rank them in descending order based on the consumed cores of the paths' server nodes.
   - Reformulate Seg^P for the top ranked physical path Eq.(2).
4. **Check if** all of the following conditions are satisfied:
   - If
   – The top ranked physical path does not currently host the SFC^r, *AND*
   – The candidate server in the top ranked physical path is different from the current server hosting SFC^r.
   – If not true Go to the next SFC^r, Step-2.
   - If
   – Utilization of the server in the top ranked physical path is higher than that of the server hosting SFC^r.
   – if not true Go to the next physical path, Step-3.
5. **- Compare** Seg^r against Seg^P using Eq.(21).
   - If all inequalities are true, then, migrate SFC^r to the top ranked physical path.
   - Update CPU and BW resources, then turn off old server node
   - Else Go to the next physical path from step-3.
6. **Go** to migrate the next SFC^r, step-2.
7. **If** no more SFC^r to migrate, **Go** back to PaNFV.

5.7. PaNFV Computational Time Complexity

Based on the size of the physical network PaNFV constructs all types of paths in $O(|N^P| + |E^P|)$ processing time, considering the total number of nodes N and edges E formulating the physical network [24]. This step is performed and saved only once before the arrival of any SFC, and it has no impact on the real computational time complexity of the RA-NFV process. However, to evaluate the computational time complexity of PaNFV, the focal computational component of the heuristic is determined based on the time consumed while sorting all the listed paths in the physical network. Accordingly, for each SFC^r the PaNFV adopted (Bubble Sort) algorithm to sort and rank all physical network paths in descending order [24], which means that PaNFV algorithm will have a quadratic computational time complexity in the order of $O(n^2)$, where n is number of potential physical paths for allocating or migrating the SFCs.

5.8. Understanding PaNFV through an illustrative example

An illustrative example is shown in Fig.2. The physical network is composed of two datacenters, four access nodes $\{a, b, c, d\}$, two router nodes $\{1, 2\}$, two switch nodes $\{3, 4\}$, and four server nodes $\{5, 6, 7, 8\}$, each server has computing capacity of 16 core units. The maximum bandwidth capacity of each edge is 20 Gbps. Since all servers are assumed empty at the beginning, then their paths will be sorted based on the server number at the middle of each path, starting from server-1 to server-4 consequently.

In this example, PaNFV works on non-consolidated mode and is required to allocate 6 SFCs as shown at the top of Fig.2. SFC-1 requesting to be allocated between access nodes a and d, demanding 2 Gbps of bandwidth, having three VNFs, where the largest of them is demanding 8 core units. PaNFV will find all paths starting and terminating with access nodes a and d, then it selects and ranks those reaching server-1 (node number-5). Next, PaNFV will select the top ranked path, which at the beginning will be path $\{a, 1, 3, 5, 3, 2, d\}$, reserve 8 core units on server node number-5, and at the same time will reserve 2 Gbps on each edge along this path from access node a to d. For SFC-2 demanding path between access nodes a to c, the largest VNF demanding 4 core units, and demanding 2 Gbps. Since server node number-5 is the most utilized among all other servers, PaNFV will rank all the paths outreaching to it and also reach to access node c, and it will select path $\{a, 1, 3, 5, 3, 2, c\}$, reserve the next available 4 core units from server node number-5, and reserve 2 Gbps along all the edges in this path. But since SFC-3 is demanding 30 Gbps, it will be rejected since PaNFV will not find any edge in the network to support it. Same case for SFC-5, PaNFV will not allocate it since the largest VNF is demanding 20 core units and no single server has such capacity.

In the case of SFC-4, the largest VNF is demanding 16 core units, PaNFV will decide to allocate them on server-2 (node number-6) through path $\{a, 1, 3, 6, 3, 2, c\}$, since server-1 does not have enough capacity to host the whole of SFC-4. Notice that PaNFV will always stick to allocate the demanded cores of any SFC on one server and will not split it between other servers at all. Finally, PaNFV will allocate SFC-6 on server-1 through path $\{a, 1, 3, 5, 3, 2, c\}$ since it is the next most utilized (server-2 is the most utilized, but it is full) server and has enough core capacity to host all of SFC-6. After each allocation attempt, PaNFV will evaluate the utilization of all server nodes in the network, and will turn them off if they are idle, such as server-3 (node number-7) and server-4 (node number-8) in datacenter 2.

5.9. Evaluation Metrics

The following evaluation metrics were used by [13] to evaluate their RCPP/RLARCDP algorithms. Accordingly, this paper used them as reference evaluation metrics as well.

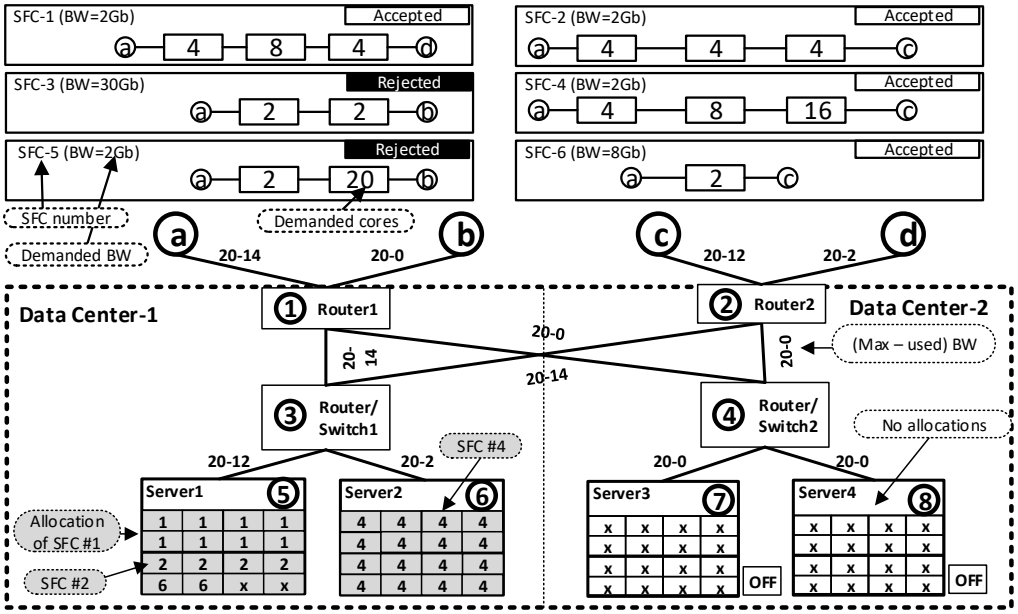

Figure 2: Numerical example about the basics of PaNFV. Six SFCs to be allocated. Each server node has computing capacity of 16 core units, and each edge has bandwidth capacity of 20 Gbps. SFC-1, SFC-2, SFC-4, and SFC-6 were allocated successfully. But SFC-3 was rejected due to exceeded bandwidth demands, and SFC-5 was also rejected due to exceeded core demands.

•Total cost, C^{tot} due to the successfully allocated SFCs :

$$C^{tot} = \sum_{\forall r \in R} (C^{pc} + C^{mig}) \tag{22}$$

•Total Cost of servers' power consumption, $C^{pc^{tot}}$ after all allocated SFCs :

$$C^{pc^{tot}} = \sum_{\forall r \in R} C^{pc} \tag{23}$$

C^{pc} is given as in Eq.(6)

•Total Cost of migrations, $C^{mig^{tot}}$:

$$C^{mig^{tot}} = \sum_{\forall r \in R} C^{mig} \tag{24}$$

C^{mig} is given as in Eq.(7)

•Fraction of dropped SFC bandwidth: is a ratio to represent how PaNFV algorithm is performing in terms of blocking rate due to limited bandwidth resources, and is calculated by dividing the blocked bandwidth demands by the SFCs over the total demanded bandwidth by all SFCs [13]:

$$\forall\, ij \in E^P \;\forall\, uv \in E^r \; Blocking = \frac{\sum_{r \in R} bw^r - \sum_{r \in R} bw^r x_{ij}^{uv^r}}{\sum_{r \in R} bw^r} \tag{25}$$

5.10. Simulation Settings for offline scenario

The simulations of PaNFV and mPaNFV were conducted against the best performing allocation and migration algorithms provided by [13] using the global policy. The same settings used to test the global policy were used for PaNFV and mPaNFV, as well as using SFCs of one VNF, and physical network topology, which includes 4 interconnected datacenters, each of them has 16 servers formed in fat-tree topology, and each server has 48 cores. The cost results of PaNFV/mPaNFV were compared to those of the global policy results in [13] when the fraction of server's idle to maximum powers was set equal to 0.3 and 1.

5.11. PaNFV and mPaNFV Performance Evaluation

To test PaNFV and mPaNFV the following sections compare their performance, against the routing and VNF placement problem algorithm (RVPP) and the revenue loss aware resource consolidation/de-consolidation problem algorithm (RLARCDP) suggested by [13]. Table 2, compares the main features of the algorithms in high level.

As an overall overview, the difference between PaNFV/mPaNFV and RCPP/RLARCDP is that PaNFV allocates the SFCs one after another on the most utilized servers, and mPaNFV checks to migrate the allocated SFCs from the least utilized servers to other highly utilized ones, then PaNFV turns off all non utilized servers afterwards. However, in [13] RVPP allocates the SFCs based on their demanded bandwidths in decreasing order and on the least stressed servers. While RLARCDP selects some of the already allocated SFCs and checks the ones that, if migrated, would minimize the network operation costs. In RLARCDP, the

Table 2: Comparing PaNFV and mPaNFV to RVPP and RLARCDP algorithms by [13]

Item	PaNFV/mPaNFV	RVPP/RLARCDP
Scenario	Offline	Offline
Goal	Minimize total cost (allocations/migration)	Minimize total cost (allocations/migration).
Strategy	Resources consolidation	Resources consolidation
Ranking	Rank paths based on utilization of their servers	Rank servers and edges based on their utilization efficiencies.
Allocating SFC's VNFs	All on one single server	All on one single server
Where to allocate VNFs	On the server of the top ranked path	On lightly loaded server with low edge stress.
Virtual edges' allocation	Together with VNFs on the top ranked path	After allocating VNFs on the shortest path.
Migration	At off-peak	At off-peak.
Migration condition	If the server's utilization below threshold	Migrate traffic from least utilized nodes.
servers' activation	One by one according to traffic load	All active with uniform loads.
Power minimization	Turn-off non utilized servers	Turn-off non utilized servers.
End-to-end delay	Yes (but not used in the simulations)	No

Table 3: Simulation settings to evaluate PaNFV and mPaNFV.

Scenario	Offline.	Network	As in Fig.7 of [13].
servers' cores	48 cores/server.	Edges' bandwidth	10 Gbps server-switch, 40 Gbps else.
SFCs topology	As in Fig.6 of [13].	Number of SFCs	500 or 1500 for blocking simulations.
SFCs bandwidth	Random (100, 150, 200, 250, 300) Mbps.	VNFs types	FW, IDS.
VNFs cores	4 cores for FW and 8 for IDS.	t_u^{proc} per VNF type	120, 160, 82.67 μsec.
Δt	1 hour.	T_{down}	2 sec.
β_d	$1.8E^{-7} - 2.7E^{-6}$	β_e	1
Simulation Runs	10	T^u	Equal to SFC bandwidth.
L^u	1500 Bytes.	P^{Busy}	1000 Watts.
P^{idle}	$a * P^{Busy}$.	a	0.3 and 1.

migrations are chosen by global policy that relies on the knowledge of the entire daily traffic profile in advance. The global policy uses cyclic-stationary traffic pattern and considers a priori chosen offline allocations.

$$\alpha_h = \begin{cases} 1, & if\ h = 0 \\ 1 - 2\dfrac{h}{Q}(1 - \alpha_{min}), & if\ h = 1, ..., \dfrac{Q}{1} \\ 1 - 2\dfrac{Q-h}{Q}(1 - \alpha_{min}), & if\ h = \dfrac{Q}{2} + 1, ..., Q - 1 \end{cases} \tag{26}$$

5.11.1. Traffic Model

To characterize semi-daily traffic profile as used by [13], it is assumed that the bandwidth demands of the SFCs are scaled according to the traffic patterns of a cycle-stationary profile, where the changes on traffic are predictable and reoccur on periodical and regular basis. Eq.(26) suggested by [13] is also used in this paper to make fair comparisons, and it provides the mathematical formula for the scale factor (α_h) controlling the proposed daily traffic profiles counted by traffic intervals h for ($Q = 24$) daily periods. The values of α_h varies over ($h = 0, 1, ..., Q - 1$), where $\alpha_0 = 1$, denotes the scale factor at peak traffic conditions, and $\alpha_{\frac{Q}{2}} = \alpha_{min}$, for the least traffic condition that triggers the migrations.

Based on this traffic profile, the ILP objective function in Eq.(5) and all the constraints starting by Eq.(8) to Eq.(20) will be checked at each traffic interval [$h \in 0, 1, ..., Q - 1$], and if satisfied, the evaluation metrics in Eq.(22) to Eq.(24) will be calculated in order to compare the results of PaNFV/mPaNFV against the results from RVPP/RLARCDP.

Table 3 lists all simulation settings used to analyze the performance of PaNFV and mPaNFV, and the results are discussed in some details in the following paragraphs.

5.12. Results and discussion

5.12.1. Total Costs

The costs of PaNFV/mPaNFV are shown in Fig.3, and the results of RVPP/RLARCDP were extracted from Fig.13 and Fig.14 in [13]. In (3a) and (3d) the total costs of PaNFV/mPaNFV were lower than those from RVPP/RLARCDP by 54% and 9% on average, noting that PaNFV/mPaNFV gave higher costs up until the cost per Gbit lost given by β_d reached $7.2E^{-7}$, where the migration costs of RLARCDP started to increase dramatically, causing the total costs of [13] to become higher than PaNFV/mPaNFV.

5.12.2. Power Consumption Costs

Regarding the results of the power consumption costs as shown in (3b) and (3e), they indicate that PaNFV consumed less power than RVPP by 55% when $a = 0.3$, but was higher than

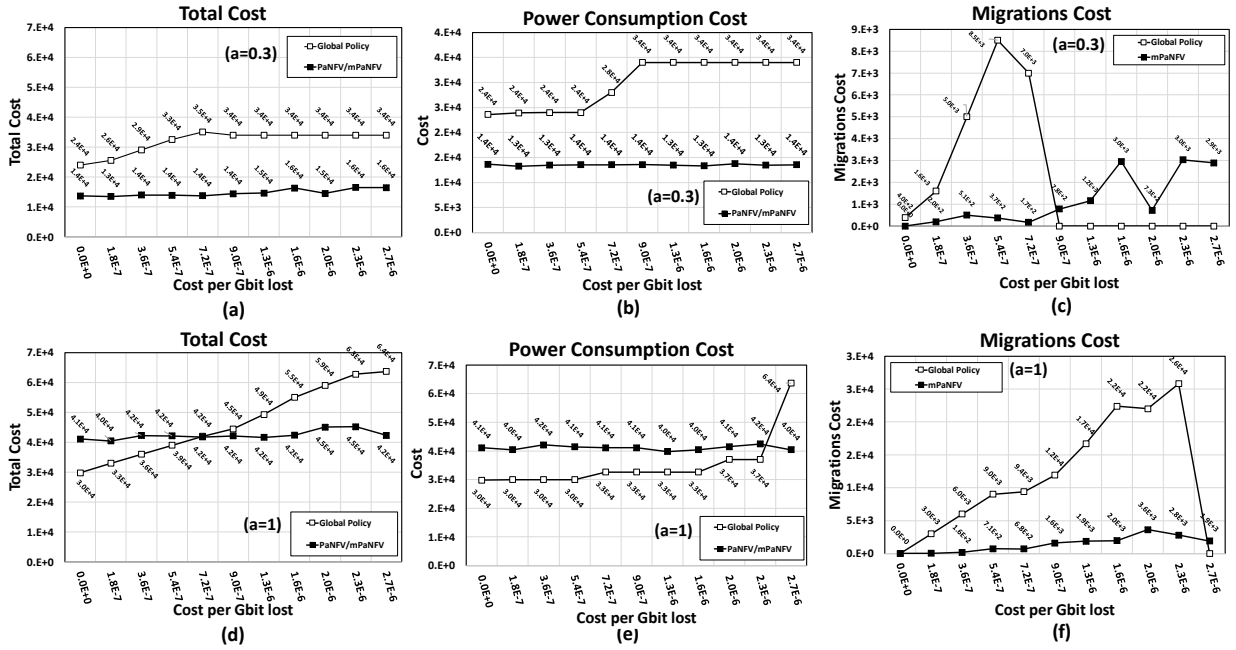

Figure 3: PaNFV/mPaNFV compared to RVPP/RLARCDP [13]. When $a = 0.3$, the average total costs of PaNFV were lower than RVPP by 54%, while power consumption costs and mPaNFV costs were lower than RVPP/RLARCDP. In (d,e,f) when $a = 1$ the average total costs of PaNFV were lower than RVPP by 9%, but its power consumption costs were higher by 16.4%, and migration algorithm mPaNFV resulted on lower values than those of RLARCDP by 88%.

RVPP by 16.4% when $a = 1$, and has nearly flat power consumptions as a function of the cost of Gbits lost. This is mainly because PaNFV does not consider future migrations in advance as the case used by RVPP, which places the virtual functions and edges of the SFCs on the least stressed servers, or directly on the servers that have available resources.

Nevertheless, the advantage of PaNFV is based on its precise allocation strategy in the first place, which will help in avoiding excessive migrations in the future. To clarify more, PaNFV carefully allocates the virtual functions and edges together using the segmentation technique, on the path of the most active (utilized) server, before activating any new servers, and keeps all other non-active ones turned off. In this way it activates the least physical resources one-by-one depending on the loads, and does not allow to use other resources unless the current active ones were fully utilized.

5.12.3. Migration Costs

Results of the migration costs are shown in (3c) and (3f), mPaNFV had almost minor reconfiguration costs, mainly because of the precise allocations by PaNFV which used least resources in the first place. However, focusing on the results of RLARCDP migration algorithm, it had much higher reconfiguration costs compared to those of mPaNFV by 43% when $a = 0.3$ and by 88% when $a = 1$, most probably because of the relaxed allocation strategy used by RVPP algorithm, causing the use of more physical resources, thus the more migrations and costs.

It is important to notice that, since RVPP allocates virtual functions first, then allocates virtual edges in another separate phase using the shortest path algorithm, that what most likely forced the migration algorithm, RLARCDP, to increase the number of migrations to minimize number of active servers, which resulted on additional migration costs, and therefore, increased the total costs.

Overall, in light of the different allocation and migration strategies used by PaNFV/mPaNFV compared to RVPP/RLARCDP, the results of PaNFV's power consumption costs were above those from the global policy on average. However, migration algorithm mPaNFV will always result on much lower costs than RLARCDP, since for the two a values, mPaNFV costs will be lower than RLARCDP by 65.5% on average. Consequently, the total costs of PaNFV/mPaNFV will be lower and much better than RVPP/RLARCDP by 32% on average.

5.12.4. Understanding Blocking Results

To evaluate the blocking probability of PaNFV, it is performance was compared against the allocation algorithm RVPP by [13]. Important to clarify that, in real world networks, SFCs should not be blocked, but can be rescheduled for future allocation attempt. However, since RVPP algorithm allow SFC blocking, and for the sake of comparison with them, blocking is also allowed by the algorithms in this paper.

The conducted simulations to test PaNFV blocking probability used the same network topology as in [13], injecting 1500 SFCs, each of a single VNF demanding 4 cores if it is a FW or EV, or 8 cores if its an IDS, and varied the demanded bandwidths randomly between $(100, 150, 200, 250, 300)$ Mbps. The results of PaNFV against RVPP placement algorithm are

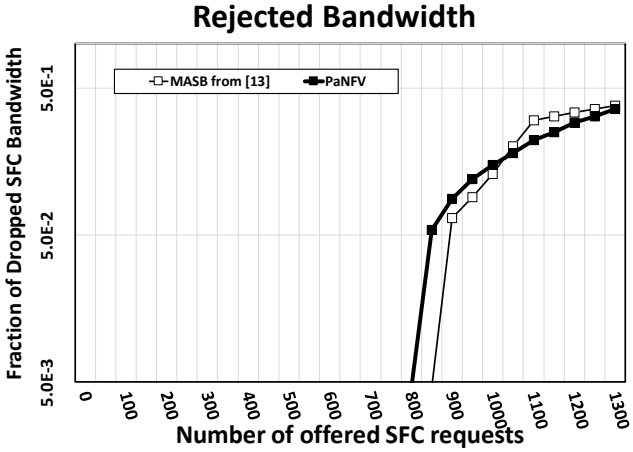

Figure 4: Fraction of Blocked bandwidth for PaNFV, RVPP from [13].

shown in Fig.4, presenting the fraction of dropped bandwidth demanded by the rejected SFCs as a function of the total demanded bandwidth by all SFCs. All algorithms, PaNFV and RVPP allocate the core resources for any VNF of any SFC on a single Server, giving that PaNFV algorithm will try to utilize the most loaded servers first before activating the new ones, but RVPP distributes the loads on the physical servers uniformly.

PaNFV uses the segmentation technique to construct the path that will host the SFC, coordinating the allocation of the virtual functions and the virtual edges together, while RVPP allocates the VNFs on the physical servers first, then uses the shortest path to allocate the virtual edges on the physical counterparts connecting the physical servers hosting the virtual functions.

From Fig.4, it can be seen that, on average PaNFV performance was better than RVPP by 7%, thanks to PaNFV's precise allocations, accepting all bandwidths until nearly 800 SFCs, then PaNFV slightly lagged behind RVPP when 800 – 1050 SFCs were allocated, but was better than RVPP Placement algorithm for SFCs between 1050 – 1500.

Overall, PaNFV allocation strategy using the segmentation technique on the most utilized paths, provided solid and stable performance close to that from RVPP, which applies vertical scaling technique to uniformly utilize the physical servers' processing and edges' bandwidth resources.

6. Online Scenario, modeling, algorithm, and simulation

6.1. Online network model and problem description

In this section, the online algorithm, oPaNFV, and the migration algorithm, mPaNFV, are introduced as a power aware coordinated NFV algorithms, allocating, migrating and removing SFCs online. It is assumed that the inter-arrival times of the SFCs reaching the physical network are normally distributed, and each SFC has arriving time and lifetime. oPaNFV allocates each SFC once it arrives and keeps monitoring until it's lifetime get expired, which would then trigger oPaNFV to terminate that SFC, by releasing its demands from their hosting physical resources in order to be used by other arriving SFCs.

Regarding migrations during online scenario, mPaNFV is inherited within oPaNFV and is basically triggered whenever the overall traffic profile in the network is at very low conditions. Moreover, mPaNFV is also kept on alert, always monitoring and evaluating the utilization of the active servers, and once it detects the utilization on one of them below the threshold migration level, mPaNFV will attempt to migrate all traffic from that low utilized server node to others, then turns it off to reduce the total power consumption in the network.

Following paragraphs explain the online objective function and its constraints, introduce the oPaNFV algorithm, and ends by discussing the simulation results.

6.2. Online Problem Formulation

6.2.1. Online objective function definition and formulation

The main target of the objective function for oPaNFV is to minimize the overall power consumption in the whole physical network, PC, by efficiently allocating the arriving SFCs, then putting into sleeping mode all idle servers that do not host any traffic. SFCs in the online scenario arrive at certain time defined by variable t^{ar}, and expire after consuming specific time units defined by variable t^{ex}. Therefore, the objective function must consider allocating these SFCs within these time boundaries.

6.3. Power Consumption Model

For the online scenario, the linear power model introduced by [22] cited by [23], is used to estimate the power consumption of the physical server, PC_{i^p}, as given in Eq.(27), which sums the server's idle power and the power consumption due to the traffic load (Utilization) on that server. Server's utilization, U_{i^p}, is a fraction between (0-1), reflecting the traffic load as a ratio between the consumed cpu_{i^p} to the maximum CPU_{i^p} on the physical node i^p given by Eq.(28).

$$\forall\ i^p \in N^P\ PC_{i^p} = P^{idle} + [P^{Busy} - P^{idle}] \times U_{i^p} \tag{27}$$

$$U_{i^p} = \frac{cpu_{i^p}}{CPU_{i^p}} \tag{28}$$

6.3.1. Online objective function definition and formulation

For the online scenario Eq.(28) will be applied as an ILP objective function, targeting to minimize the total power consumption in the whole network after each allocated SFC. It will be constrained by capacity and domain constraints to guarantee that the optimal solution should be reached within the demanded time intervals by the SFCs.

To solve the objective function as an ILP problem, variable $x_{i^p}^{u^t}$ is used, which takes a binary value of 1 if server i^p is assigned to host SFC^r during the time interval $t \in [t^{ar}, t^{ex}]$, 0 otherwise. Variable $\lambda_{i^p}^{t}$ is 1 if the physical node i^p is active, or 0 if it is turned-off during time interval t. Variable $\mu_{i^p}^{u^t}$ is 1 if during time interval t, all VNF nodes hosted by i^p are migrating, or 0 if they remain. Variable $mig_{ij}^{uv^{r^t}}$ is set to 1 if virtual edge (u, v) hosted by the physical edge (i, j) is migrating, variable mig^{r^t}

is set to 1 if SFC^r is migrating, and variable $x_{ij}^{uv^{r^t}}$ is set to 1 if physical edge (i, j) is hosting virtual edge (u, v).

The objective function is shown in Eq.(29):

$\forall\ t \in [t^{ar}, t^{ex}]$

$$\min PC = \sum_{\forall i^p \in N^P} (P^{idle} + [P^{Busy} - P^{idle}] \times U_{i^p}^t) \quad (29)$$

6.3.2. Constraints formulation

1. *Constraints on server nodes*

$$\forall\ i^p \in N^P\ \forall\ t \in [t^{ar}, t^{ex}] \sum_{r \in R, u \in N^r} cpu_{u^t} x_{i^p}^{u^t} \leq CPU_{i^p} \quad (30)$$

$$\forall\ i^p \in N^P,\ \forall\ t \in [t^{ar}, t^{ex}] \sum_{u \in N^r} x_{i^p}^{u^t} \geq \lambda_{i^p}^t \quad (31)$$

2. *Constraints on physical edges*

$$\forall\ ij \in E^P\ \forall\ uv \in E^r\ t \in [t^{ar}, t^{ex}] \sum_{r \in R} bw^{r^t} x_{ij}^{uv^{r^t}} \leq BW_{ij} \quad (32)$$

$$r \in R\ t \in [t^{ar}, t^{ex}] \sum_{ij \in P_{sd}} d_{ij}^t \leq d^{r^t} \quad (33)$$

3. *Migration constraints*

$$\forall\ i^p \in N^P\ \forall\ t \in [t^{ar}, t^{ex}]$$

$$\sum_{r \in R, u \in N^r} cpu_{u^t} x_{i^p}^{u^t} \leq cpu^{min} \quad (34)$$

$$\forall\ i^p \in N^P\ \forall\ t \in [t^{ar}, t^{ex}]$$

$$\sum_{r \in R, v \in N^r} cpu_{v^t} x_{i^p}^{v^t} + \sum_{r \in R, u \in N^r} cpu_{u^t} mig_{i^p}^{u^t} \leq CPU_{i^p} \quad (35)$$

$$\forall\ ij \in E^P\ t \in [t^{ar}, t^{ex}]$$

$$\sum_{r \in R, \forall\ wz \in E^r} bw^{r^t} x_{ij}^{wz^{r^t}} + \sum_{r \in R, \forall\ uv \in E^r} bw^{r^t} mig_{ij}^{uv^{r^t}} \leq BW_{ij} \quad (36)$$

$$\forall\ r \in R\ t \in [t^{ar}, t^{ex}]$$

$$\sum_{\forall\ (i,j) \in P_{sd}} d_{ij} \leq d^{r^t} mig^{r^t} \quad (37)$$

4. *Domain constraints*
   Similar to Eq.(16)-Eq.(20).

The above constraints behave similarly as when they were used for the offline scenario. However, the main difference in the online scenario, is that each constraint must also be true in each time interval $t \in [t^{ar}, t^{ex}]$.

6.4. Optimal solution of the online objective function

The optimal solution to solve the online RA-NFV objective function in Eq.(29), implies introducing binary constraints to allocate all VNFs on one physical server node, which is similar to the multi-dimensional NP-Hard Bin Packing problem that is not solvable in polynomial times [8][9]. Therefore, this paper proposes a modified version of the online power aware allocation algorithm developed by [1] called oPaNFV, to solve the online objective function, and fully coordinating allocating the VNF nodes and edges together on the best physical path.

6.5. oPaNFV explained

The oPaNFV pseudo-code is shown in Algorithm 3. In the general sense it works similar to the online version from [1], but for NFV environment. oPaNFV has four main steps as in the offline version PaNFV, namely, the initialization, segmentation, allocation, and the updating, where each of them works exactly as they did for the PaNFV. As a summery, at each time interval t, if an SFC arrives the network at t^{ar}, oPaNFV will construct its segment as in Eq.(3), and will formulate the physical segment for the top ranked physical path P_{sd} as in Eq.(4). Then oPaNFV will check if physical path has enough resources to host the arriving SFC^r during time interval $t \in [t^{ar}, t^{ex}]$. If yes, oPaNFV will allocate that SFC^r on the selected physical path and update the whole network accordingly.

To make oPaNFV work for online conditions as those in 5G networks, it includes a new part to monitor the lifetimes of the arriving SFCs, as well as to handle the termination procedure if any SFC is expiring. It works as follows, at each time unit the algorithm keeps checking if the lifetime of any SFC is about to be expired, oPaNFV will then recall the path hosting that SFC, and remove its demands from the hosting resources of that path. In parallel with that, and after each removal cycle, oPaNFV will always keep evaluating the power consumption of each server node if it is idle or loaded, and turns it off if it does not host any traffic to save the overall power consumption in the whole network accordingly.

Algorithm 3, oPaNFV Pseudo-Code

1. Input: G^P and G^r.
2. While t≠0 **Do**
3. **Follow steps-(2-7)** as PaNFV in Algorithm-1
4. **For** all allocated SFCs **Do**
   - Check if any SFC is expiring
   - Release the demands of the expiring SFCs (cpu_u and bw^r) from the hosting resources.
5. **Follow steps-(8-11)** as PaNFV in Algorithm-1

Important to highlight that, the most significant addition by oPaNFV is the inclusion of migrations during online. At every moment of time oPaNFV will keep evaluating the utilization levels in the physical network, and if the three migration conditions (as clarified in mPaNFV) are met, oPaNFV will trigger the migration procedure, and accordingly will turn off any non

Table 4: Simulation settings to test oPaNFV

Common		**Exp-3**	**Impacts of loading network**.
SFCs' lifetime	400 time units in all experiments.	Main objective	Test long/short SFCs' lifetimes.
Migration	Always active in all experiments.	Expiring SFCs	$(1-10)$ every $(5-10)$ time units.(Long)
Arriving SFCs	$(5-10)$ every $(5-10)$ time units.		$(5-10)$ every $(5-10)$ time units.(Short)
Consolidation	Non-consolidation mode.	Simulation time	5000 time units.
Exp-1	**Impact of migrations on oPaNFV**.	**Exp-4**	**Impacts of delay**.
Main objective	oPaNFV with/out migrations.	Main objective	oPaNFV with/out delay.
Expiring SFCs	$(1-10)$ every $(5-10)$ time units.	Expiring SFCs	$(1-10)$ every $(5-10)$ time units.
Simulation time	$100,000$ time units.	Simulation time	5000 time units.
Exp-2	**Impact of VNFs' consolidations**.	Physical delay	Random $(0.1-1)$ msec per edge.
Main objective	oPaNFV with/out consolidations.	SFC delay	Random $(1-20)$ msec, less sensitive SFCs.
Arriving SFCs	10 every 10 time units.		Random $(0.1-5)$ msec, Sensitive SFCs.
Expiring SFCs	10 every 10 time units.		
Simulation time	5000 time units.		

loaded server to minimize the total power consumption. Notice that, oPaNFV can also trigger the migration algorithm for maintenance and emergency conditions at any time interval t.

6.6. Simulation Settings to test oPaNFV

Because oPaNFV is an evolution of the offline algorithm PaNFV, which was evaluated against [13], then in the online scenario, all settings listed in Table 3 are also valid and used during the evaluation procedure of oPaNFV, including the same physical network topology and SFCs' configurations. In addition to that, since [13] does not include online algorithms, the authors could not compare their results to them, but they modified the online algorithms from [1] to work for NFV frameworks and migrations including end-to-end delay.

Four different experiments were conducted to evaluate the impacts of migrations, VNFs' consolidations, SFCs' lifetimes, and delay on the overall performance of oPaNFV. Simulation settings for all these experiments are listed in Table 4, wich starts by summarizing the common parameters that were used in all experiments unless otherwise stated, assuming that the average lifetime for any SFC is 400 time units, migrations are always allowed, and the number of arriving SFCs selected randomly between $(5-10)$ SFCs every $(5-10)$ time units. Moreover, notice that in all experiments the types of SFCs are similar to those from [13], where the number of VNFs per SFC are randomly selected between $(1-3)$, but the VNFs will be allocated following the non-consolidation mode as follows: if the arriving SFC has one VNF, then 4 cores are allocated, while if the SFC is of two VNFs $4, 8$, then 8 cores are allocated, and if it is of three VNFs $4, 8, 4$, also 8 cores will be allocated only.

The first experiment (Exp-1) does not include end-to-end delay, since it targeted evaluating the basic performance of oPaNFV with and without migrations, in terms of power consumption, utilizations of the physical servers, as well as the number of active servers. In Exp-1, a total of $(5-10)$ SFCs were assumed randomly arriving the physical network every $(5-10)$ time units, while $(1-10)$ SFCs assumed expiring every $(5-10)$ time units. The migration algorithm, mPaNFV on Exp-1 was always kept active, and depending on the migration conditions as defined in Algorithm 2, the migrations could be triggered at anytime during the whole simulation time. All demanded bandwidth values for the arriving SFC were randomly selected between $(100, 150, 200, 250, 300)$ Mbps, and the simulations were conducted over $(0-100,000)$ time units.

Second experiment (Exp-2) targeted evaluating the total power consumption performance of oPaNFV and other metrics, when the VNFs' consolidations on the physical servers were not allowed, and compare that to the results when they were allowed. All other simulation settings are similar to Exp-1, including always active migrations, except that the simulation time was set equal to 5000 time units, assuming that 10 SFCs arriving, and other 10 expiring every 10 time units, and Exp-2 does not include end-to-end delay too.

Third experiment (Exp-3) targeted evaluating the performance of oPaNFV when the network is loaded, by varying the number of expiring SFCs, and analyzing how is that impacting the physical network. The experiment was conducted when the average lifetime per arriving SFCs was varied by setting the lifetimes equal to $400 + \delta^l$, giving that δ^l was selected random between $(1-10)$ time units for longer lifetimes (i.e. heavily loaded network), and between $(5-10)$ time units for shorter lifetimes (i.e. lightly loaded network).

Fourth experiment (Exp-4) targeted evaluating the performance of oPaNFV when end-to-end delay was used. Generally, it is possible to include any type of delay in the segmentation technique, giving that the authors did extensive review to the possible delays including propagation, transmission, processing and queuing delays, and concluded to follow the recommended end-to-end delay values for 5G technologies as stated in [2], which includes all delay types. Accordingly, to simulate impacts of end-to-end delay on the allocations of SFCs on $5G$ like networks, Exp-4 will inject two types of SFCs to be allocated on the physical network, and evaluate the resultant end-to-end delays by comparing the results of both types to each other. The first type includes SFCs which are less sensitive to physical network delay, as is the case for most $5G$ applications, and the second type of SFCs are those very sensitive to end-to-end delay. Delays in the physical network edges were randomly se-

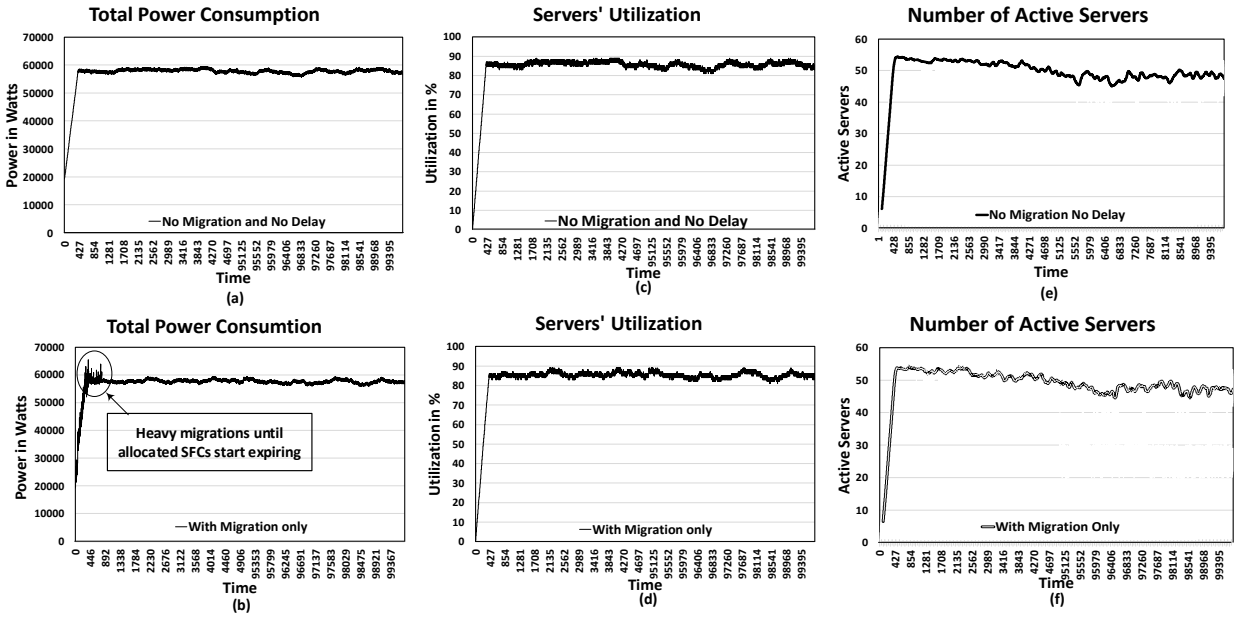

Figure 5: Overall Performance of oPaNFV with and without migrations. Total power consumptions of oPaNFV in (a) without migrations, and in (b) with migrations, showing limited migrations in the early stages of the simulation time. Servers' utilizations in (c) and (d), and average active servers in (e) and (f) were high, mainly due to the long lifetimes of the arriving SFCs.

lected between $(0.1 - 1)$ msec, and the demanded end-to-end delay by the first type of SFCs were randomly selected between $(1 - 20)$ msec, and between $(0.1 - 5)$ msec for the second type of SFCs.

6.7. Discussing results of oPaNFV performance

6.7.1. Impacts of migrations on oPaNFV

Simulation results of Exp-1 are shown in Fig.4, presenting the performance of oPaNFV with and without migrations. The experiment was conducted for $100,000$ time units, but due to the large number of samples, the results are shown for the first and last 5000 time units using the real numbers from the simulations. Moreover, to show some details and to give real sense about the overall behavior of the online algorithm for a large number of demands, the results of the migration case were drawn in separate figures than those without migration.

The total power consumptions are shown in Fig.4a when migration was not included, and in Fig.4b with migrations, indicating that oPaNFV conducted some limited migrations in the early stages of the simulation time. However, the behavior of the algorithm stabilized when the allocated SFCs started expiring after passing the first 400 time units, where the total power consumption remains smoothly varying on average around 57.04 KWatts without migrations, representing (89.12%) of the network's maximum power, and 57.16 KWatts (89.31%) with migrations. These results reflect the online nature of oPaNFV and how it handles the arriving and expiring SFCs, with almost very negligible migrations along the simulation time frame. Servers' utilizations were in the averages of 83.98% when no migrations were allowed as in Fig.4c, and 83.62% with migrations Fig.4d.

Important to notice that the high values of power consumption and utilization were directly affected by the longer lifetimes of the arriving SFC, which is an indication about the overall network loads and the resources consumptions. Lastly, in Fig.4e and Fig.4f the average number of active servers was high, around 49 in both cases, mainly due to the long lifetimes of the arriving SFCs, and that is why the total power consumptions were high too.

The overall conclusion from Exp-1 is that the performance of oPaNFV on a long run simulation times is stable and very smooth, but greatly depends on the lifetimes of the arriving SFCs. However, the bold conclusion from the previous observations is that, giving the careful allocation strategy of oPaNFV, migrations in online are mostly very minor or negligible, almost very few migrations at the early stage of the experiment. Therefore, it would be more efficient to use the migrations for maintenance or emergency cases, than to minimize the total power consumption in the physical network. This will further reduce the complexity in oPaNFV and make it even much faster.

6.7.2. Impacts of VNFs consolidations

Fig.5 shows the results of oPaNFV when VNFs' consolidations were allowed and not allowed. Remember, the consolidate case is when the reserved cores on the server are the sum of the total demanded cores by all VNFs of the SFC, and the non-consolidated is when the cores were reserved based on the demands of the largest VNF in the SFC. Note that it is only in Exp-2 the results were drawn using the moving average trends to show the curves more clearly.

To understand the overall impacts on oPaNFV, Fig.5a shows that the average power consumption of the whole physical net-

work for the two cases were very close to each other. It was around 56.27 KWatts, that represents 87.92% of the network's maximum power in the case of non-consolidated VNFs, compared to 58.76 KWatts (91.81%) with VNFs allowed to consolidate.

The resulted savings in the total power consumption were around 4% on average, which means that when non-consolidation technique is used, the physical network will have additional room to allocate more SFCs, but at slightly lower power consumption rates than the consolidated case.

However, edges' utilizations results shown in Fig.5b did not follow the same trend as the previous results from other metrics, and they were 6.7% in the case of non-consolidated VNFs and 4.1% when consolidations were allowed. In non-consolidated VNFs case, the higher values in the edges' utilizations were due to oPaNFV's strategy of using the same paths more frequently for multiple SFCs until their servers get fully utilized. However, in the case of consolidated VNFs the servers will be utilized faster and their paths will be less used by multiple servers, accordingly, they will be less utilized. To some degree this is confirmed by the number of active servers, which were around 50 in the non-consolidated case and 49 with consolidations as shown in Fig.5c.

The overall conclusion from experiment-2 is that for some virtual network services, where the VNFs of a specific SFC are strictly ordered and not working on parallel, non-consolidation mode could be a good choice, since it allows the physical network to accommodate more SFCs at lower power consumption rates. However, if the VNFs are not ordered and my demand to allocate them in parallel, the consolidated mode could be applied, noting that it will slightly increase the total power consumption.

6.7.3. Impacts of varying SFCs' lifetimes

To understand the impacts of SFCs' lifetimes on the load of the physical network, Exp-3 varied the lifetimes of the arriving SFCs in two separate runs, long versus short living SFCs. For long living SFCs, the lifetimes were varied between (400 + {1 *to* 10})) time units, and for short living SFCs it varied their lifetimes between (400 + {5 *to* 10})) time units. As shown in Fig.6a, when less number of SFCs were leaving (i.e. has longer lifetimes), the total power consumption in the whole network was almost hovering around 61.64 KWatts, representing 96.3% of the maximum network's power, but it was around 39.68 KWatts (62.1%) when more SFCs were leaving (i.e. having shorter lifetimes). Same trends can be seen for the servers' utilizations, which were around 93.56% and 44.65% as in Fig.6b. In addition, when considering the number of active servers in each case, Fig.6c shows that on average, 25 servers remain active when SFCs of the shorter lifetimes were hosted, compared to 58 active server when SFCs of longer lifetimes were allocated.

The overall conclusion from Fig.6 is that the longer the lifetimes of the allocated SFCs, the more the power consumption, and the higher the servers and edges' utilizations on the physical network. Most noticeable, is that even for a slight variation in the number of expiring SFCs, the impacts on the physical network metrics were immense and so obvious. Suggesting that, among all other physical network's parameters, SFCs' lifetime would most likely be the dominating factor on the overall performance of the physical network.

6.7.4. Impact of end-to-end delay on the resource allocation process for 5G networks

As a main contribution from this paper, Exp-4 was designed to reflect the most realistic performance of oPaNFV, including end-to-end delay as a main constraint during the allocation process, in addition to keeping migrations as always active all the time. Delay results are shown in Fig.7, presneting the impacts on the overall performance of oPaNFV when less sensitive to delay SFCs were allocated compared to those having more strict and aggressive delay settings.

As a major evaluation metric about the impacts of delay on the resource allocation problem using oPaNFV, Fig.7a shows that the acceptance ratios were on the averages of 89.8% for the less sensitive to delay applications, compared to a reduced acceptance ratios in averages of 70.47% for the sensitive to delay SFCs, consequently, notice that in Fig.7b the total power consumption when sensitive to delay SFCs were allocated resulted on 53% lower consumptions than the less sensitive to delay case, mainly because of the less utilized resources, and less accepted demands. Moreover, Fig.7c show impacts on servers' utilizations when the less and the more sensitive to delay SFCs were allocated through oPaNFV, which recorded average values of 93.21% and 20.51%. The low values in the aggressive delay conditions were mainly due to the low numbers of the allocated SFCs.

Overall, results from Exp-4 underline the importance of including delay in the allocation process, also clearly point-out that, considering delay will have some serious consequences on the performance of the physical network, basically due to the physical characteristics and limited number of sufficient paths with accepted delay.

However, for 5G applications, still these delay results are not accepted and need to be enhanced big time to reach the minimum acceptance ratios hopefully above 98 – 99%. Moreover, the lower acceptance ratios in the aggressive scenario reflect that, much more additional enhancements are needed to support such sensitive to delay applications in the physical network layer to live up to 5G delay requirements.

Unfortunately, these are extremely hard problems to solve from the physical network point view. Nevertheless, using smart allocation algorithms such as oPaNFV, combined with edge computing, small mobile cells' designs, in addition to some advanced features, could be promising to enhance these acceptance ratios.

6.7.5. oPaNFV allocations' times for the SFCs

The allocations' times shown in Fig.8 are oPaNFV consumed time in milliseconds to embed each SFC, when the settings of first experiment (Exp-1) were applied. oPaNFV took on average 60.5519 msec to allocate a single SFC, giving that, the code was developed using Eclipse IDE for Java Developers, version

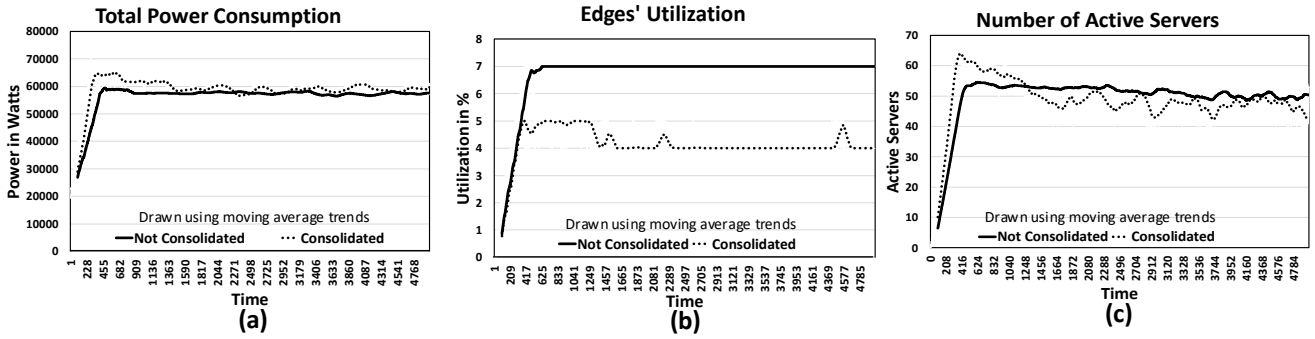

Figure 6: oPaNFV performance with/out consolidating the VNFs. Referring to (a), average power consumptions for non-consolidated VNFs were lower than the results of the consolidated case, but edges' average utilizations shown in (b) were higher than the consolidated experiment. (c) shows the average number of active servers where high, confirming the power consumption values.

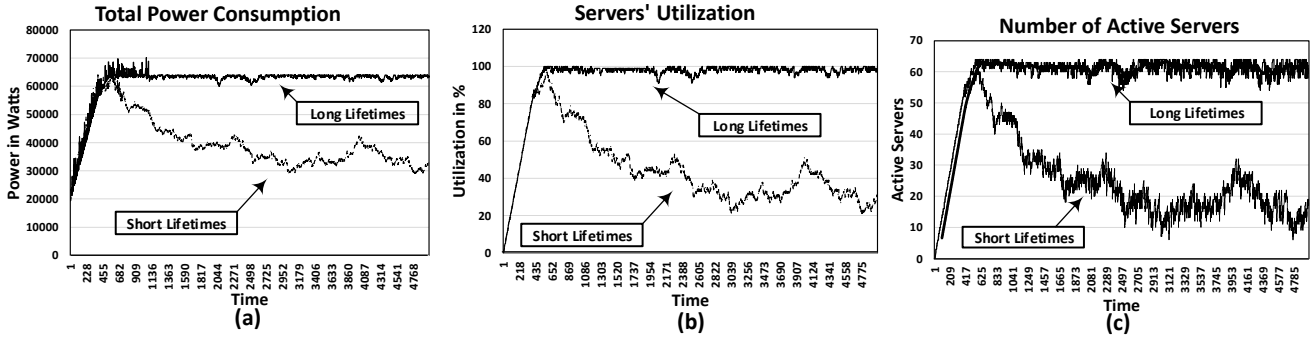

Figure 7: oPaNFV performance varying lifetimes. (a) shows that SFCs of longer lifetimes consumed more power than the SFCs of shorter lifetimes, same trends for servers' utilizations in (b), and (c) shows that, when SFCs of longer lifetimes were allocated, more servers remain active.

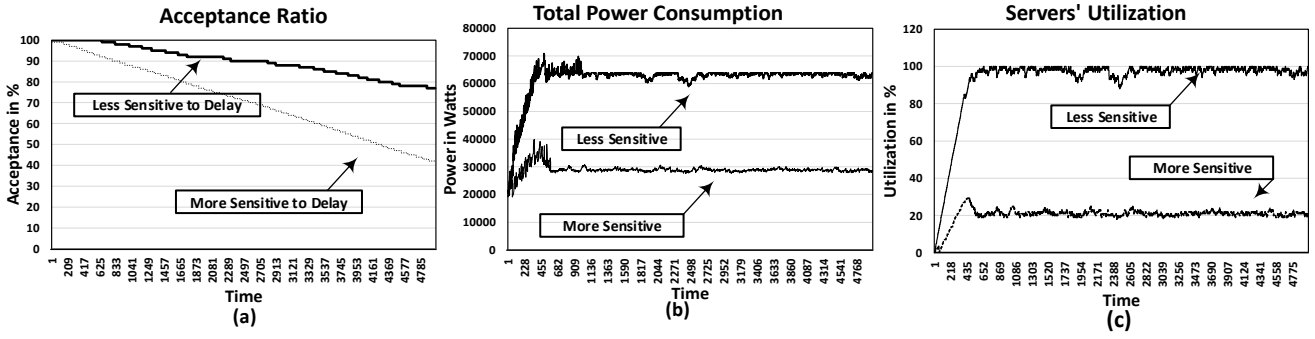

Figure 8: oPaNFV performance varying delay. In (a) shows acceptance ratios when allocating the less sensitive to delay SFCs. (b) shows that allocations of the less sensitive to delay SFCs consumed more power, and same trends can be seen for the results of servers' utilizations in (c).

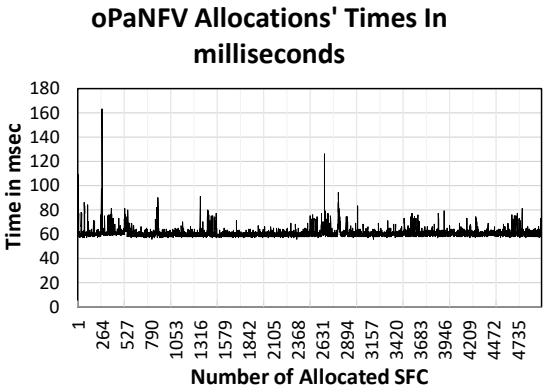

Figure 9: oPaNFV Embedding Time in milliseconds.

Mars.2 Release (4.5.2). For all conducted experiments in this paper, the used machine had an i7 processor, 5820K, 12 cores, 3.30 GHz, 16 GB RAM, and the operating system was Ubuntu 16.04.3 LTS.

6.8. Future work

In future work the authors are gearing towards developing another version of the segmentation methodology, supporting resource allocation in the core network and the more dynamic access network as well. In addition to try other traffic models, topology types, and configurations as well. Also, the authors suggest to try online migrations for very large networks, including long living allocation demands.

7. Conclusions

This paper proposed an offline and online power aware algorithms to solve the resource allocation problem within the frameworks of NFV, and to minimize the overall costs and power consumption rates in the physical network. Each of the algorithms is integrated with a migration strategy to further minimize the power consumption. The simulation results for the offline algorithm showed that on average it had lower total costs and lower migration cost by 32% and 65.5% compared to the state-of-art. On the other hand, the online algorithm managed to allocate the Network Services in average times of 60 msec, in addition to that, it had very negligible migrations, mainly due to the efficient and careful allocations of the online algorithm in the first place. This means that, when using the proposed online algorithms in this paper, migrations in the online scenario will most likely be used for maintenance or emergency conditions.

Moreover, when demands of longer lifetimes were allocated, the overall performance of the online algorithm showed that, the power consumption in the physical network will on average increase by 34.2% more than the case of allocating demands of shorter lifetimes, suggesting to consider some trade-offs when allocating the demands. Finally, when aggressive end-to-end delays where applied, as a demonstration for the real world 5G applications, the performance of the online algorithms in terms of acceptance ratio degraded by 19.3%, suggesting the significant implications of including delay during the resource allocation process.

8. Acknowledgment

This work has been partially supported by the Ministerio de Economy Competitividad of the Spanish Government under project TEC2016-76795-C6-1-R and AEI/FEDER, UE, and the SGR project, grant number 2017 SGR 397, from the Generalitat de Catalunya.

References

[1] Khaled Hejja, Xavier Hesselbach, 2018. "Power aware coordinated virtual network embedding with 5G delay constraint," Journal of Network and Computer Applications, Elsevier, 2018. DOI:10.1016/j.jnca.2018.10.005.

[2] ITU-T Focus Group, 2017. "IMT-2020 Deliverables," www.itu.int.

[3] 5G Americas, 2018. "5G Americas White Paper: Cellular V2X Communications Towards 5G," www.5gamericas.org

[4] ITU, 2011. "ITU-T Y.3001: SERIES Y: Global Information Infrastructure, Internet protocol Aspects and Next-Generation Networks. Future networks: Objectives and design goals," www.itu.int.

[5] 3GPP TR 28.801 (V15.0.0), 2017. "Study on management and orchestration of network slicing for next generation network," portal.3gpp.org

[6] ETSI GS NFV 004 v1.1.1, 2013. "Network Function Virtualisation (NFV); Virtualization Requirements," www.etsi.org

[7] ETSI GS NFV 002 v1.1.1, 2013. "Network Function Virtualisation (NFV); Architectural Framework," www.etsi.org

[8] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. De Turck and R. Boutaba, 2016. "Network Function Virtualization: State-of-the-Art and Research Challenges," in IEEE Communications Surveys and Tutorials, vol. 18, no. 1, pp. 236-262. https://ieeexplore.ieee.org/document/7243304/

[9] J. Gil Herrera and J. F. Botero, 2016. "Resource Allocation in NFV: A Comprehensive Survey," in IEEE Transactions on Network and Service Managment, vol. 13, no. 3, pp. 518-532. DOI:10.1109/TNSM.2016.2598420

[10] J. F. Riera, X. Hesselbach, M. Zotkiewicz, M. Szostak and J. F. Botero, 2015. "Modeling the NFV forwarding graph for an optimal network service deployment," 2015 17th International Conference on Transparent Optical Networks (ICTON), Budapest, pp. 1-4. DOI:10.1109/ICTON.2015.7193483

[11] J. F. Riera, X. Hesselbach, E. Escalona, J. A. Garcia-Espin and E. Grasa, 2014. "On the complex scheduling formulation of virtual network functions over optical networks," 2014 16th International Conference on Transparent Optical Networks (ICTON), Graz, pp. 1-5. DOI:10.1109/ICTON.2014.6876564

[12] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba and O. C. M. B. Duarte, 2016. "Orchestrating Virtualized Network Functions," in IEEE Transactions on Network and Service Management, vol. 13, no. 4, pp. 725-739. DOI:10.1109/TNSM.2016.2569020

[13] V. Eramo, E. Miucci, M. Ammar and F. G. Lavacca, 2017. "An Approach for Service Function Chain Routing and Virtual Function Network Instance Migration in Network Function Virtualization Architectures," in IEEE-ACM Transactions on Networking, vol. 25, no. 4, pp. 2008-2025. DOI:10.1109/TNET.2017.2668470

[14] C. Pham, N. H. Tran, S. Ren, W. Saad and C. S. Hong, 2017. "Traffic-aware and Energy-efficient vNF Placement for Service Chaining: Joint Sampling and Matching Approach," in IEEE Transactions on Services Computing. DOI:10.1109/TSC.2017.2671867

[15] O. Soualah, M. Mechtri, C. Ghribi and D. Zeghlache, 2017. "Energy Efficient Algorithm for VNF Placement and Chaining," 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Madrid, pp. 579-588. https://ieeexplore.ieee.org/document/7973745/

[16] Kim, S., Park, S., Kim, Y., 2017. "VNF-EQ: dynamic placement of virtual network functions for energy efficiency and QoS guarantee in NFV," Cluster Computing, vol 20, issue 3, pp 2107-2117. DOI:10.1007/s10586-017-1004-3

[17] B. Kar, E. H. K. Wu and Y. D. Lin, 2017. "Energy Cost Optimization in Dynamic Placement of Virtualized Network Function Chains," in IEEE Transactions on Network and Service Management, vol. PP, no. 99, pp.372-386. DOI:10.1109/TNSM.2017.2782370

[18] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos and L. P. Gaspary, 2015. "Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions," 2015 IFIP-IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, 2015, pp. 98-106. DOI:10.1109/INM.2015.7140281

[19] V. S. Reddy, A. Baumgartner and T. Bauschert, 2016. "Robust embedding of VNF/service chains with delay bounds," 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Palo Alto, CA, pp. 93-99.DOI:10.1109/NFV-SDN.2016.7919482

[20] L. Qu, C. Assi and K. Shaban, 2016. "Delay-Aware Scheduling and Resource Optimization With Network Function Virtualization," in IEEE Transactions on Communications, vol. 64, no. 9, pp. 3746-3758. 2016.DOI:10.1109/TCOMM.2016.2580150

[21] Bhamare, Mohammed Samaka, Aiman Erbad, Raj Jain, Lav Gupta, H. Anthony Chan, 2017. "Optimal virtual network function placement in multi-cloud service function chaining architecture," Computer Communications, Volume 102, 2017, Pages 1-16, ISSN 0140-3664, DOI:10.1016/j.comcom.2017.02.011

[22] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso, 2007. "Power provisioning for a warehouse-sized computer," In Proceedings of the 34th annual international symposium on Computer architecture (ISCA '07). ACM, New York, NY, USA, 13-23.DOI:10.1145/1250662.1250665

[23] M. Dayarathna, Y. Wen and R. Fan, 2016. "Data Center Energy Consumption Modeling: A Survey," in IEEE Communications Surveys and Tutorials, vol. 18, no. 1, pp. 732-794. DOI:10.1109/COMST.2015.2481183

[24] J. Kleinberg and E. Tardos, 2009. "Algorithms Design," Addison-Wesley.

[25] J. Qin, Y. Wu, Y. Chen, K. Xue and D. S. L. Wei, "Online User Distribution-Aware Virtual Machine Re-Deployment and Live Migration in SDN-Based Data Centers," in IEEE Access, vol. 7, pp. 11152-11164, 2019. DOI:10.1109/ACCESS.2019.2891115

[26] Esfandiarpoor, Ali Pahlavan, Maziar Goudarzi, "Structure-aware online virtual machine consolidation for datacenter energy improvement in cloud computing," Computers and Electrical Engineering, Volume 42, 2015, Pages 74-89. DOI:10.1016/j.compeleceng.2014.09.005

[27] Rachael Shaw, Enda Howley, Enda Barrett, "An energy efficient anti-correlated virtual machine placement algorithm using resource usage predictions," Simulation Modeling Practice and Theory, Volume 93, 2019, Pages 322-342. DOI:10.1016/j.simpat.2018.09.019

[28] Ranjbari, Javad Akbari Torkestani, "A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers," Journal of Parallel and Distributed Computing, Volume 113, 2018, Pages 55-62. DOI:10.1016/j.jpdc.2017.10.009

[29] Zhihua Li, Chengyu Yan, Lei Yu, Xinrong Yu, "Energy-aware and multi-resource overload probability constraint-based virtual machine dynamic consolidation method," Future Generation Computer Systems, Volume 80, 2018, Pages 139-156. DOI:10.1016/j.future.2017.09.075